AN INFORMATION-THEORETIC PERSPECTIVE FOR LEARNING SYSTEMS WITH ENGINEERING APPLICATIONS

BY

CHUAN WANG

A DISSERTATION PRESENTED TO THE GRADUATE SCHOOL
OF THE UNIVERSITY OF FLORIDA IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

UNIVERSITY OF FOLRIDA

1996

ACKNOWLEDGMENTS

First and foremost, I would like to thank my thesis advisor, Dr. Jose C. Principe, who has illustrated to me what it means to do brilliant research, and equally importantly, how to explain it to others. Most important, he granted me a great degree of freedom to pursue my own course of thought through the duration of my doctoral research. His style of supervision was perfectly suited to my style of investigation.

Special thanks go to Dr. John Harris for his constructive comments, discussions, and friendship. I also thank my committee members Dr. Jian Li, from whom I learned a lot of fresh knowledge about spectral estimation; Dr. William Edmonson who helped me understand the depth of adaptive filters and global optimization algorithms; and Dr. Mark Yang who made a lot of statical concepts more clear to me. I also wish to thank Dr. H. Latchman and Dr. Sencer Yerelan of Industrial and Systems Engineering department for their special help during the exams of the thesis.

My friends, Dr. Jyh-Ming Kuo and Dr. Samel Celebi, offered me a lot of stimulating discussions during the course of my study, especially in the beginning. I thank all the membership of the CNEL Lab at the University of Florida for providing a work environment that was both congenial and stimulating.

Several students have directly contributed to the work presented here. Hsiao-Chun Wu performed most experiments on blind sources separation given in Chapter 4, Li-Kang Yen did the simulation for transform domain filter of Chapter 3, and my friend Chang Chun-Ming at Medical Image Processing Group of Computer Science Department did the wavelet decomposition presented in Chapter 3. Craig Fancourt and Dongxin Xu helped me to get speech signals from SwitchBoard and TIMIT databases and provided very useful

discussions about speech processing. Chen-Hsien Wu repeatedly provided me with computer support and friendship.

Finally, I thank my parents, my wife, my brothers, and sister for their consistent love, support and encouragement, which made this thesis possible.

TABLE OF CONTENTS

Abstract of Dissertation Presented to the Graduate School
of the University of Florida in Partial Fulfillment of the
Requirements for the Degree of Doctor of Philosophy

AN INFORMATION-THEORETIC PERSPECTIVE FOR LEARNING SYSTEMS WITH ENGINEERING APPLICATIONS

By

Chuan Wang

August 1996

Chairman: Dr. Jose C. Principe
Major Department: Electrical and Computer Engineering

The major goal of this study is aimed at building a unifying perspective for most learning systems (adaptive filters and neural networks). A detailed analysis of the adaptation rules is presented from the point of view of generalized correlation learning. The analysis also reveals that learning in recurrent networks is equivalent to learning with second-order correlation with different time lags, which highlights why recurrent systems extract time information. It is well known that supervised systems can be used either for static learning (functional mapping MLPs) or temporal learning (time delay neural networks or the Gamma model). But in unsupervised learning, almost all neural networks are trained statistically due to the absence of a teacher signal. Therefore, a unified perspective of temporal supervised and unsupervised learning requires a mathematical extension to unsupervised learning. The focus of extending static unsupervised systems to temporal learning will be made with the Principal Components Analysis (PCA) network. PCA is one of the dominant networks in the unsupervised family and it is based on the Hebbian rule which

plays, by itself a fundamental role for unsupervised learning. PCA in time is examined in detail. It is shown that PCA in time gives a set of adaptive time-varying orthogonal basis ordered by variance which constitute the signal sub-space. The relationships between PCA in time, Fourier analysis, and wavelets are also pointed out. An application to subspace adaptive filtering is outlined which decreases significantly the training time. Then, as an application of the PCA concepts to time processing, a neural topology to compute the crosscorrelation and autocorrelation on-line is proposed. The algorithm exploits the unifying perspective developed for the learning rules based on correlation learning. This network is then used for blind sources separation, which is a difficult problem because the solution must estimate the transfer function of a linear system based on the outputs of the system alone. We then turn to the other goal of the thesis---to propose a unified perspective for both supervised and unsupervised learning. A simple but poorly understood relationship between supervised and unsupervised learning is revealed. It is shown that when the desired signal is a zero mean noise, the supervised learning is statistically equivalent to unsupervised learning. This result combined with the knowledge of autoassociative learning provides a basis to present a perspective for learning from the point of view of information theory. The main theoretical conclusion of the thesis can be outlined as: In a supervised learning system, when the mutual information between the input and the desired signal reaches its extreme (maximum or minimum) the learning degenerates into an unsupervised paradigm. With this perspective, the classification of learning in supervised or unsupervised is not only based on the existence of a desired signal but must also take into consideration the relationship between the external signals.

CHAPTER 1

INTRODUCTION

1.1 Learning Process

During the past three decades there has been a mounting interest in adaptive systems, especially neural networks. Among the many interesting properties of a neural network, the property that is of primary significance is the ability of the network to learn from its environment, and to improve its performance through learning; the improvement in performance takes place over time in accordance with some prescribed measure. A neural network learns about its environment through an iterative process of adjustments applied to its synaptic weights. Ideally, the network becomes more knowledgable about it environment after each iteration of the learning process.

There are too many notions associated with learning to justify defining the term in a precise manner [Haykin, 1994]. The concept of learning is so vast that it is difficult to agree on a precise definition of the term. For example, learning viewed by a psychologist is quite different from learning in a classroom sense, while the learning concepts in machine intelligence are different from adaptive systems, i.e. symbolic and numerical learning are very different. Recognizing that our particular interest is in adaptive neural networks, we use a definition of learning that is adapted from Mendel [Mendel, 1970; Haykin, 1994]: Learning is a process by which the free parameters of a neural network are adapted through a continuing process of stimulation by the environment in which the network is embedded. The type of learning is determined by the manner in which the param-

eter adjustments take place.

The environment associated with learning usually means the external signals such as input and desired signals.

This definition of the learning implies the following sequence of events:

(1) The neural network is stimulated by an environment.

(2) The neural network undergoes changes as a result of this stimulation.

(3) The neural networks responds in a new way to the environment, because of the changes that have occurred in its internal structure.

To be specific, consider a pair of node signals x_j and v_k, which generated from environment, connected by a weight w_{kj}, as depicted in Figure 1, where **u** and **d** represent the environment stimulations.

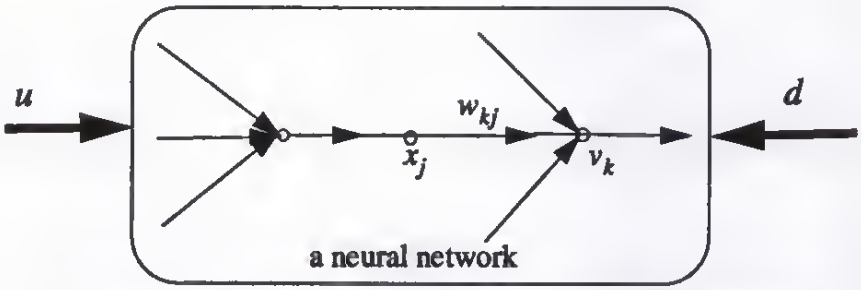


Figure 1. A signal flow graph of learning in a neural network

Signal x_j represents the output of neuron j, and signal v_k represents the internal activity of neuron k. In the context of weight w_{kj}, the signals x_j and v_k are commonly referred to as presynaptic and postsynaptic activities, respectively. Mathematically, iterative learning process is expressed as the change of the synaptic weights w_{kj}; that is,

$$w_{kj}(n+1) = w_{kj}(n) + \nabla w_{kj}(n) \tag{1}$$

where $w_{kj}(n)$ denotes the value of the weight at time n, $\nabla w_{kj}(n)$ is an adjustment term.

Eq. (1) sums up the overall effect of events (1) and (2) implicit in the definition of the learning process presented above. In particular, the adjustment $\nabla w_{kj}(n)$ is computed as a result of stimulation by the environment (event (1)), and the updated value $w_{kj}(n+1)$ defines the change made in the network as a result of this stimulation (event (2)). Event 3 takes place when the response of the new network, operating with the updated set of parameters $w_{kj}(n+1)$ is reevaluated.

A prescribed set of well-defined rules for the solution of a learning problem is called a learning algorithm. As one would expect, there is no unique learning algorithm for the design of learning systems. Rather, we have a "tool kit" represented by a diverse variety of learning algorithms, each of which offers advantages of its own. Basically, learning algorithms differ from each other in the way in which the adjustment $\nabla w_{kj}(n)$ to the weights is formulated. Another consideration is the manner in which a neural network (learning machine) relates to its environment. In the latter context, we speak of a learning paradigm referring to a model of the environment in which the neural network operates. This thesis will concentrate on building a unified perspective the learning paradigms, by relating in supervised and unsupervised learning. The reinforcement learning, which is also considered as branch of learning paradigms [Haykin, 1994], is not considered in this study due to its immaturity.

1.2 Motivation of the Research and the Outline of the Thesis

It is well known that learning algorithms are the most important aspect of adaptive systems, although the network topology also plays an important role in performance. Based on the learning algorithms, adaptive systems can be divided into one of two learning paradigms---supervised and unsupervised [Hertz et al, 1991]. Examples of supervised learning algorithms include Backpropagation (BP) [Rumelhart et al, 1986], Least Mean Square (LMS) [Windrow and Hoff, 1960] which can be viewed as a special case of BP for

linear networks, and some other modified forms of LMS such as recursive least square (RLS), leaky LMS, and normalized LMS, etc. [Haykin, 1991]. Unsupervised learning usually includes the Hebbian rule [Hebb, 1949], anti-Hebbian rule [Rubner and Tavan,1989], Oja's rule [Oja, 1982], Sanger's rule [Sanger, 1989], competitive learning [Rumelhart and Zipser, 1985], and self-organizing maps (SOMs) [Kohonen, 1990]. Hebbian rule is the most important rule since it serves as a basis for all the other rules. The acknowledged distinction between these two kinds of learning is based on whether a teacher (desired) signal is used in the formulation. In learning with supervision, it is traditionally assumed that the desired response for the learning system is known in advance, and the difference between the desired and the actual response, that is, the error, is used to correct its behavior [Tsypkin, 1971]. In unsupervised learning, the desired response of the learning system is not explicitly known, and thus we cannot directly formulate and use the error of the learning system in order to improve its behavior. But the interaction among system components and the environment sets up an implicit learning rule and the system does self-learning based on this underlying rule [Mendel et al., 1970]. It is generally accepted that the supervised and the unsupervised learning are totally different learning methods and have different system structures [Haykin, 1994]. Although a lot of approaches and results have been given in each individual area, only a handful of research work investigated the relationship between these two learning paradigms. Another important point is that the learning algorithms are closely associated with the learning paradigms. For example, BP and LMS are used for supervised learning, whereas Hebbian and anti-Hebbian are assigned to unsupervised learning. To the best of our knowledge, there is no unified perspective available to study these different rules, which motivated this study

The importance and advantage of unifying these relationships is multi-fold. First, it will elucidate the connection between both learning paradigms, which will be very helpful to understand the learning mechanisms. Second, it will explicitly show the role that each signal plays in the system. Third, it will provide a direction on how to build new algo-

rithms and architectures for adaptive systems in which supervised and unsupervised learning are merged together. Finally, it will strengthen the point that supervised learning based on gradient information is also biological plausible.

Although there is no unifying perspective to analyze learning in adaptive systems from the point of view of algorithmic structure, recent results are providing some clues. Correlation has been recognized in the unsupervised paradigm for a long time [Haykin, 1994] and gradient information is extensively used in unsupervised learning such as PCA learning and competitive learning [Oja, 1992]. However, correlation learning was not outlined as a generalized learning principle of supervised learning using gradient information. The implicit connection between supervised and unsupervised learning has also been studied from different angles. Baldi revealed that multilayer linear network can be used to extract principal components when the desired signal is the input itself, which is called auto-association [Baldi and Hornik, 1989]. Bourlard showed that this conclusion is also valid for nonlinear MLPs [Bourlard and Kamp, 1988]. Kosko pointed out that all BP, LMS, Hebbian and Adaptive Vector Quantization algorithms use gradient descent or gradient ascent learning algorithms [Kosko, 1990]. Another clue is through the use of to use information theoretic concepts. Information theory, which describes transformation of information in a system, is well known since the seminal work of Shannon [Shannon, 1949]. The big advantage of information theory is that it does not work for a specific signal, but instead uses probability space which provides a unifying point to study a system. Hence, it is very natural to use information theory to describe a learning system. Although Linsker [Linsker, 1986], Plumbley [Plumbley and Fallside, 1988] applied information theory to unsupervised learning and other researchers used mutual information and cross-entropy error criteria in supervised learning [Hinton, 1989; Richard and Lippmann, 1991], only a few papers used information theory to study the relationship between supervised and unsupervised learning. Nadal was able to show that the maximum information that can be utilized by supervised learning is equal to the maximum information that can be

transmitted by the dual network, trained with the unsupervised model [Nadal and Parga 1994].

Therefore, we can conclude that the relationship between supervised and unsupervised learning is still an open problem.

Since the goal of any theoretic study aims at providing solutions to practical problems, it is very significant to identify important, still unsolved problems to which the new learning techniques to be developed can be applied. As for the applications of the adaptive neural approach to be developed in this work, we are going to focus on blind sources separation problem. Blind separation problems are met in communication, speech signal collection and processing, sonar signal processing and so on. Although several methods based on higher-order statistic and adaptive filter approaches such as the Bussgang algorithm have been reported [see Haykin, 1994 for a comprehensive review], the problem is still unsolved. Recently, some approaches based on neural networks have been proposed. Jutten and Herault proposed a nonlinear network to separate independent signals [Jutten and Herault, 1991]; Bell and Sejnowski applied the maximum entropy method to separate mixed signals [Bell and Sejnowski, 1995]; Matsuoka et al. minimized the correlation between the output signals [Matsuoka et al., 1995]. We will propose a different approach based on our study on the temporal PCA network and correlation learning principle. And experimental results show that the proposed algorithm is as good as others for the blind sources separation problem, but much simple to implement.

The core of this study is composed of analyzing various learning systems from different angles: unifying learning rules for both supervised and unsupervised systems, relationship between supervised and unsupervised learning, PCA learning in time, and their applications. Because of the distinct character of these individual topics, rather than organizing the thesis in the conventional way, where there is a separated results section, we prefer to present the results in the body of each chapter followed by some discussions.

The flow of the thesis can be summarized as follows: Chapter 2 deals with the uni-

versal learning principle for both supervised and unsupervised systems. At first, it points out that the fundamental rules such as Hebbian, anti-Hebbian, and competitive learning are correlation based learning in which a learning rule can be decomposed into several terms which show the explicit correlation relation between variables. Since the correlation learning seems to be the basic adaptation mechanism in the brain [von Malsburg, 1994]. It is important to reveal correlation learning in artificial learning systems. It has been found, through a detailed analysis of each individual learning algorithm, that supervised learning with gradient information of mean square error criterion leads to the correlation rule too, but the correlation here is a combination of the basic correlation rules such as Hebbian and anti-Hebbian. Furthermore, it has been also found that learning in dynamic systems such as recurrent neural networks employes correlation terms with different time lags, which gives another evidence to show the power of recurrent system to process temporal signal. Based on the above observations, a universal correlation rule is outlined as follows: The dominant learning principle for most adaptive systems (supervised or unsupervised) can be viewed as a combination of simple correlation learning rule. Furthermore, this combination is a linear superposition operation with time-varying coefficients. The theoretic background behind this conclusion is that the backward network for the optimization using gradient based mean square error cost function is a time-varying coefficients linear system. This chapter also reveals the function each variable plays in a learning system, which is critical for us to design novel learning algorithms.

In Chapter 3, the static PCA learning is extended to time sequences. This is a necessary and important step to build a unified perspective for learning systems since the temporal processing abilities of most unsupervised systems have not been studied. However, supervised learning has been applied to both static and temporal processing. First of all, a tapped delay line structure is proposed for temporal PCA. The training algorithm used for the temporal structure is still the Hebbian rule. It will be shown that the temporal PCA network provides a set of time-varying basis for the signal sub-space. And these basis func-

tions are adjusted based on the second-order statistics of the input signal. Furthermore, it will be shown mathematically that these basis functions will converge to complex exponential functions when the input signal is a stationary signal, and these bases provide information of the input signal in both time and frequency domains for locally stationary signal with a selected window size. Moreover, relationships between PCA in time and filter banks and eigenfilter are discussed, which are helpful to understand the PCA in time network, since eigenfilter and filter bank have been used in signal processing for many years. Some examples of PCA in time learning are given so as to show the time-frequency and on-line orthogonalization characteristics of the temporal PCA learning.

In Chapter 4, a topology to compute the crosscorrelation matrix of two signals is proposed. The algorithm combines the characteristics of both supervised and unsupervised learning, and therefore is called teacher forcing learning. First, the estimation of temporal crosscorrelation between two signals use an FIR (Finite Impulse Response) structure. Then, an IIR (Infinite Impulse Response) structure using the Gamma memory is shown to be able to compute the crosscorrelation function in an efficient way. Moreover, we extend the FIR structure to the nonlinear case in which high-order statistics of the training signals are employed implicitly. The method can be viewed as an estimation of the nonlinear crosscorrelation between two signals. The algorithm is applied to the blind separation problem. It is shown that it can be used for Gaussian and non-Gaussian, as well as stationary and nonstationary signals. The separation results are compared with other approaches and show the advantages of the proposed algorithm.

In Chapter 5, on the basis of generalized correlation learning rule presented in Chapter 2, we go to the details of the relationship between supervised and unsupervised systems. A simple relationship is pointed out in this chapter. It is shown that the distinction between supervised and unsupervised learning is not on whether the desired signal is presented or not. It depends strongly on the correlation between the external signals such as input and desired signals. If these two signals are statistical independent, the underlying

learning is actually unsupervised, as well as if the two signal are exactly the same. Moreover, we address this relationship from the point of view of information theory. We proposed to use the mutual information concept to describe a learning system, and show that when the mutual information between the input and desired signals reaches its extreme (maximum or minimum). The learning system is an unsupervised system. Otherwise, it is a supervised system. With these conclusions, a unified perspective based on information theory is presented at the end of this chapter.

Applications, simulation results, and discussions are provided at the end of each chapter to show the usefulness and valid of the approaches and algorithms presented there.

Finally, conclusions of the study and the possible future research issues are given in Chapter 6. We will recapitulate the main results obtained in this thesis from the global point of view to summarize the importance for new algorithms design, better understanding of learning systems, and novel network structures design. At the same time, we would like to highlight some possible further research directions related to this thesis.

CHAPTER 2

CORRELATION LEARNING OVERVIEW: A GENERALIZED PRINCIPLE FOR LEARNING SYSTEMS

2.1. Introduction

It is well known that learning rules of neural networks are classified into supervised and unsupervised learning. If an external signal serves as a teacher signal, the learning is named as supervised learning. Otherwise, it is called unsupervised learning [Haykin, 1994; Hertz et al., 1991]. Although these two learning paradigms are suitable for most systems, they are applied differently. In supervised systems, the backpropagation (BP) algorithm is basically used for static multi-layers perceptron (MLP), real time recurrent learning (RTRL) algorithm is specified to recurrent network with arbitrary connections, and perceptron learning algorithm is applied to simple perceptron; in unsupervised systems, besides the Hebbian rule which is biologically plausible [Hebb, 1949], principal components analysis (PCA), and competitive learning rule are also very important. Although a lot of research has been done in the above mentioned learning algorithms, we are still seeking the common factors among them. In other words, the fundamental question of learning "what are the principles according to which local synapse changes are organized in order to yield global learning of complex behaviors for the organism?" remains largely unanswered [Baldi, 1995]. Two key ideas, as pointed out by Baldi [Baldi, 1995], are playing an important role in learning systems. One is the Hebbian rule which is the basis for most unsupervised learning such as PCA and competitive learning. Another idea is the gradient descent technique which constitutes the perspective for most super-

vised learning such as multilayer perceptrons, recurrent networks, and radial basis networks. The big advantage of gradient descent is that it is very efficient in practice due to the stability of numerical computation and simple mathematics.

We will point out that correlation learning plays a fundamental role in learning systems in this chapter. But first, some basic concepts and definitions of correlation for second-order and higher-order statistics are reviewed.

2.2. Some Basic Concepts and Definitions

In this section, some basic concepts and definition related to correlation are given, which will be useful for us to understand the results in the subsequent sections.

Definitions of second-order covariance and correlation [Picinbono, 1993]: Consider two random signals x (n) and y(n), which are complex. The following function defines the covariance between these two signals:

$$\gamma(t,\tau) = E[x(t)\,y(t+\tau)] - m_x(t)\,m_y(t+\tau) \tag{2}$$

where E is the expectation operator, t and τ are the discrete time indices, and m_x and m_y are mean corresponding to random process x(t) and y(t), respectively. Note that when the mean m_x and m_y are zero, the covariance function degenerates to correlation function, and when x(t)=y(t) $\gamma(t,\tau)$ is named autocorrelation, otherwise it is called crosscorrelation. Since the means are constants for any given stationary random signals, we shall use the correlation in this thesis to represent the function defined in Eq. (2).

The covariance or correlation are measurements of the second-order statistics. Similarly, higher-order statistics can be defined, which will be used in higher-order neural networks.

Definition of higher-order correlation [Mendel, 1991]: Let

$X(t) = col[x_1(t), x_2(t), ..., x_p(t)]$ be a collection of random processes $x_1(t), x_2(t), ..., x_p(t)$, the cumulants for the second- and third-order are defined by:

$$C_{x_i,x_j}(t,\tau) = E[x_i(t)\,x_j(t+\tau)] \tag{3}$$

$$C_{x_i,x_j,x_k}(t,\tau_1,\tau_2) = E[x_i(t)\,x_j(t+\tau_1)\,x_k(t+\tau_2)] \tag{4}$$

where t, τ, and τ_i, i=1,2, are time indices. Similarity, the fourth-order cumulant and higher-order cumulants can be defined [Swami and Mendel, 1990]. Notice that the second-order cumulant is just the correlation function given by Eq. (2). Thus, cumulants are generalized correlation.

It should be mentioned that for random variables, the above definitions are still true if the time indices are dropped. We will use the correlation for both random variables and random processes.

We will use abusively the term correlation to express either second-order correlation and covariance since all of them describe some kinds of correlation between random processes.

2.3. Correlation Learning is a Universal Principle for Learning Systems

We will now focus on particular systems and show that the correlation learning principle plays a dominant role. Based on the discussions given in section 1.1, we know that all information from the given training data is collected by the weights. Therefore, the adjustment rule of the weights represents the characteristics of learning. It should be pointed out that correlation learning is not a new concept. It has been used for a long time in unsupervised systems. The main purpose of this chapter is not to propose the correlation learning, but to show the different forms of correlation learning in existing supervised

and unsupervised learning systems. We will first review the correlation learning for several unsupervised systems in section 2.3.1 and section 2.3.2. Then, the correlation learning is extended to supervised learning in section 2.3.3.

2.3. 1 Hebbian Rule is Correlation Learning

The off-line Hebbian rule for a linear single layer static network can be mathematically expressed as the adjustment applied to the synapse weight w_{kj} is in the form [Haykin, 1994]

$$\Delta w_{kj} = \eta E[y_k x_j] \tag{5}$$

where η is a small positive constant that determines the rate of learning.

Obviously, the adjustment of the weight given in Eq. (5) represents the correlation between the data y_k and x_j. Eq. (5) can only be used for batch learning since the accumulation of samples are necessary to estimate the expectation operation. Normally the expectation operator can be substituted by a average operator A[.], defined by

$$A[x] = \lim_{L \to \infty} \frac{1}{L} \sum_{i=1}^{L} x_i \tag{6}$$

Then, the on-line learning algorithm can be approximated simply by dropping the average operator [Widrow and Stearns, 1985]:

$$\Delta w_{kj} = \eta y_k x_j \tag{7}$$

In this thesis, the adjustment term is usually presented in the on-line form, and the

expectation operator will be omitted in most cases. We will emphasize this point only if necessary.

The nonlinear Hebbian rule usually takes the form of [Oja and Karhunen, 1993; Karhunen and Joutsensalo, 1995],

$$\Delta w_{kj} = \eta g(y_k) h(x_j) \tag{8}$$

where g(.) and h(.) are nonlinear functions.

It is important to note that a random variable passing through a nonlinear device is still a random variable, but with different statistics. Obviously, Eq. (8) emphasizes the correlation between two new random variables $g(y_k)$ and $h(x_j)$ which are nonlinear functions of the original signals y_k and x_j, respectively. Therefore, we conclude that either linear Hebbian rule or nonlinear Hebbian rule is correlation learning. Most of unsupervised learning algorithms are based on the basic Hebbian rule as we will see below.

2.3.2 Unsupervised Learnings Based on Hebbian Rule are Correlation Learnings

We will review several typical learning algorithms such as PCA, competitive learning, and linear associative memory and show that they are correlation learning.

(i) PCA learning

A PCA network, which is a linear single layer network ($y_k = \sum_{j=1}^{n} w_{kj} x_j$, k=1,..,m, and j=1,..,n), is given in Figure 2.

In PCA learning, the adjustment of weight is based on the Hebbian rule given by Eq. (7) but with a normalization on the weights since the Hebbian rule in Eq. (7) is an unstable rule [Oja, 1982]. Hence, the adjustment of the synapse weight is in the form [Oja, 1982]

$$\Delta w_{kj} = \eta y_k [x_j - y_k w_{kj}] \tag{9}$$

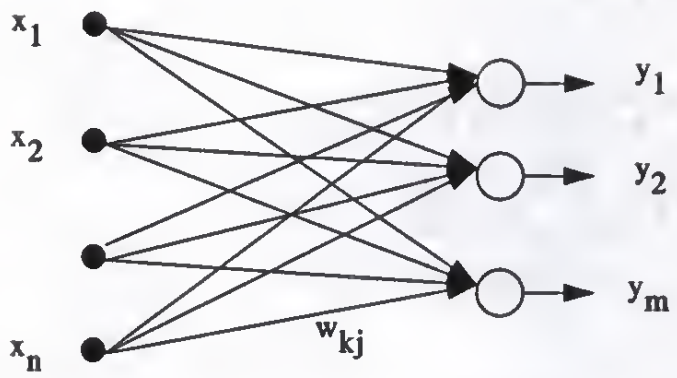


Figure 2. A linear network for the PCA

Eq. (9) can be rewritten into

$$\Delta w_{kj} = \eta y_k x_j - \eta y_k y_k w_{kj} \tag{10}$$

The first factor in the right side of Eq. (10) is the Hebbian rule of Eq. (7), and the second term is the crosscorrelation function between the output signal y_k and its projection on the weight vector $y_k w_{kj}$. Hence, the PCA learning can be viewed as a normalized correlation learning.

(ii) Competitive learning

The competitive learning algorithm guarantees that only one network output, or only one per group, fires at a time. The output units compete, and are therefore often called winner-take-all units [Hertz et al., 1991].

In order to show explicitly the correlation in the competitive learning, a new look of the competitive learning from the supervised learning viewpoint will be given next.

Figure 3 shows a linear network for competitive learning.

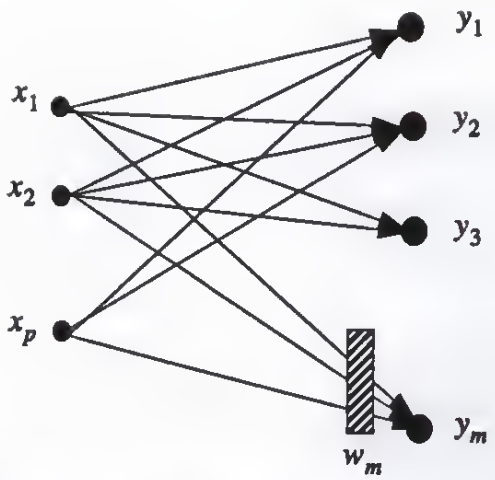


Figure 3. A linear network for competitive learning

The learning procedure can be summarized as:

(1) Computing the distances, $||e_i||$, i=1,..,m, between the input $x = [x_1, x_2, \ldots, x_p]$ and the weight vectors w_i, i=1,...,m.

(2) Finding the minimum distance, $||e_{i,min}||$ and its corresponding weight vector $w_{i,min}$

(3) Adjusting w_{min} by

$$\Delta w_{i,min} = \eta (x - w_{i,min}) \tag{11}$$

where η is step size.

(4) The rest of the weight vectors are fixed.

(5) If the $e_i^2 \leq \Upsilon$ which is a preset bound, stop learning, otherwise,

(6) Go back to step (1) to iterate the procedure.

It was observed that there is a cost function associated with the learning rule given in Eq.(11). The cost function can be defined by [Ritter and Shulten, 1988]:

$$J(w) = \min_{w} \frac{1}{2} ||x - w_{i,min}|| \tag{12}$$

where || || represents Euclidean norm.

Applying gradient technique to the cost function, the adaptation rule in Eq. (11) can be obtained for the winning neuron.

Traditionally, the competitive learning shown in Figure 3 is studied as an unsupervised learning which means no desired signal is used to guide the learning. In order to formulate the competitive learning into a supervised paradigm, let us compare the adaptation rule of the weight between the competitive learning and classical LMS algorithms for linear transversal filter (tapped delay line).

It is well known that the adaptation rule of the classical LMS for a linear transversal filter has the form of [Widrow and Stearns, 1985]:

$$\Delta w_{i,min} = 2\eta\varepsilon_i u \tag{13}$$

where the error is defined by $\varepsilon_i = (d_i - y_i)$ for the desired signal d_i and output y_i, and u is the input signal.

If we define, in Eq. (13), the error $\varepsilon_i = x - w_{i,min}$ and set the input u=1, this adjustment and Eq. (11) have the same form. Hence, the unsupervised competitive learning can be expressed in a supervised paradigm as given in Figure 4, where w_i, i=1,..,m, is a 1-by-p vector.

The input of the network given in Figure 4 is a indicator matrix $X = [\mathbf{0}, \ldots, \delta, \ldots, \mathbf{0}]$, with one column at the ith position which corresponds the weight vector of fired neuron and other zeros, and where $\mathbf{0}$ represents a vector with all elements zeros, and $\delta = 1$ is a vector with all elements ones.

Up to now, we have seen that the competitive learning can be viewed a supervised learning with an indicator matrix X. And now we discuss how to set the X. In the learning procedure,

$$e^2_{min} \le e^2_{(2)} \le \ldots \le e^2_{max} \tag{14}$$

where $e^2_{(2)}$ is the error with second smallest value. The ordering can be also integrated into Figure 4.

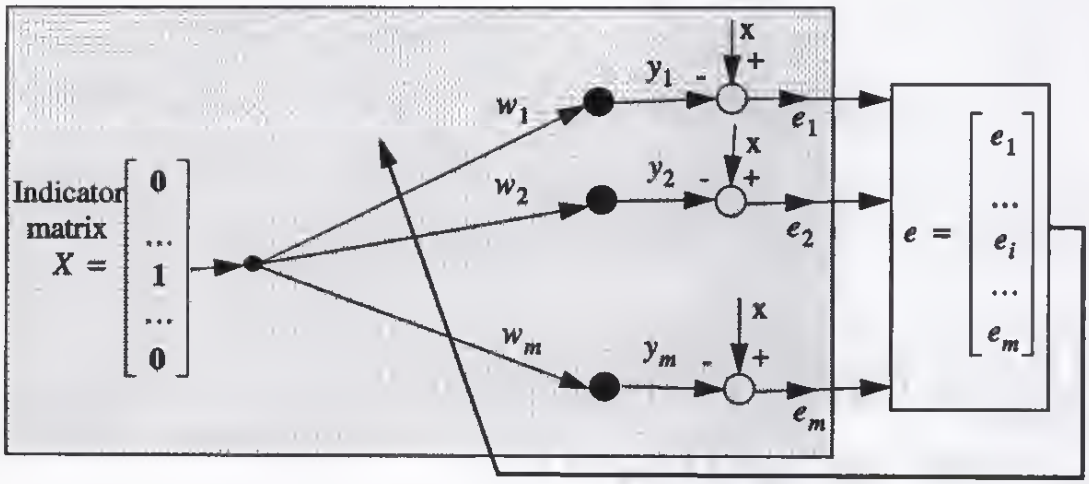


Figure 4. Supervised version of the competitive learning

Therefore, the adaptation rule for each fired neuron can be written into

$$\Delta w_i = \eta \delta (x - w_i) \tag{15}$$

The correlation relationship in Eq. (15) is clear since only in the direction specified by δ the input and weight vector are useful.

(iii) Linear associative memory

The linear associative memory (LAM), which is also called correlation matrix, has distinct characteristics showing that this class of learning system uses the correlation directly. LAMs can be implemented with two types of association: autoassociation and heteroassociation. In an autoassociative memory, a key vector is associated with itself.

Accordingly, the input and output signal spaces have the same dimensionality. In a heteroassociative memory, on the other hand, arbitrary key vectors are associated with other arbitrary vectors. The output space dimensionality may or may not be equal to the input space dimensionality and the output is substituted by desired response.

The block diagram of the linear associative memory is given in Figure 5.

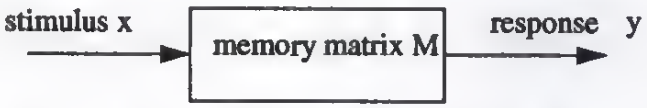


Figure 5. The block diagram of the linear associative memory

where x and y are the data vectors.

We assume there are q pairs of data patterns x_k and y_k, k=1,..,q, for data vectors x and y. It can be shown, for these q pairs of associated patterns, that the adjustment to the memory matrix W takes the form of [Haykin, 1994],

$$\Delta w_k = y_k x_k^T \tag{16}$$

where T is the transpose operator. The term $y_k x_k^T$ represents the outer product of the key pattern x_k^T and the memorized pattern y_k. Hence, it is clear that the linear associative memory follows the correlation learning principle.

Based on the discussions given in this subsection, we see that most unsupervised systems are correlation learning.

2.3. 3 Supervised Learning is a Combination of Correlation Learning

In subsections 2.3.1 and 2.3.2, it was clear that not only the Hebbian rule but also other unsupervised rules such as competitive learning are correlation learning. Here, we

will study supervised learning, which is depicted in Figure 6, where the desired signal and the input signal are denoted by d and x, respectively. Supervised learning occurs in the form of LMS, BP, real-time recurrent learning, and higher-order learning rules. Moreover, we will point out that those learning algorithms are also biologically plausible with the results obtained in this section. For other learning algorithms such as backpropagation learning rule, least mean square (LMS), and learning rule for radial basis function networks will be skipped since all of them can be viewed as special cases of the BP rule.

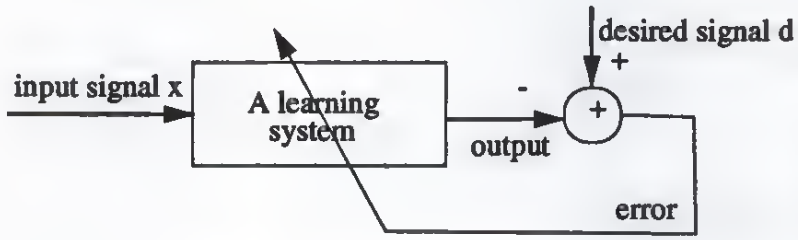


Figure 6. A supervised learning system

(i) BP learning

Backpropagation rule is a remarkable achievement in the development of neural networks. Without loss of generality, we analyze the BP algorithms using a two-layer multilayer perceptron which is depicted in Figure 7.

Following the discussion given by Hertz, let us first define some variables [Hertz et. al, 1991].

Given Pattern α, hidden unit k receives a net input

$$h_k^{\alpha} = \sum_j V_{kj} x_j^{\alpha} \tag{17}$$

and produces the output (activation of the hidden layer)

$$P_k^\alpha = f(h_k^\alpha) = f(\sum_j V_{kj} x_j^\alpha) \tag{18}$$

output unit i thus receives

$$h_i^\alpha = \sum_k W_{ik} P_k^\alpha = \sum_k W_{ik} f(\sum_j V_{kj} x_j^\alpha) \tag{19}$$

and produces for the final output

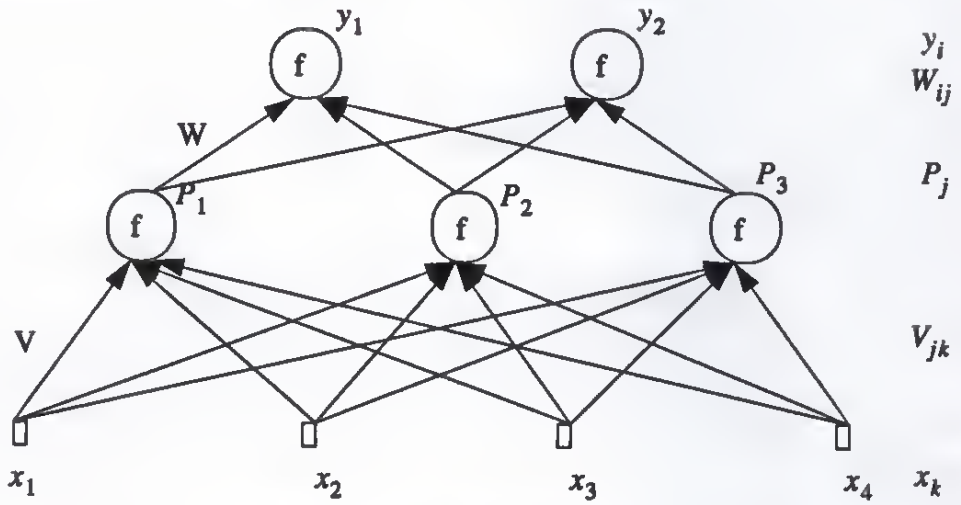


Figure 7. A two layers MLP

$$y_i^\alpha = f(h_i^\alpha) = f(\sum_k W_{ik} P_k^\alpha) = f(\sum_k W_{ik} f(\sum_j V_{kj} x_j^\alpha)) \tag{20}$$

Then, using the mean square error (MSE) criterion and gradient decent method, we get [Hertz et al., 1991]

(1) Weights between the output layer and hidden layer are adjusted in the following way

$$\Delta W_{ik} = \eta \varsigma_i^\alpha P_k^\alpha \tag{21}$$

where we have defined (d_i is the desired signal corresponding to y_i)

$$\varsigma_i^\alpha = f'(h_i^\alpha)\,[d_i(t) - y_i(t)] \tag{22}$$

(2) Weights of other layers are adjusted by

$$\Delta V_{kj} = \eta \varsigma_k^\alpha x_j^\alpha \tag{23}$$

with

$$\varsigma_k^\alpha = f'(h_k^\alpha) \sum_i W_{ik} \varsigma_i^\alpha \tag{24}$$

where the expectation operator is done with respect to all patterns.

Decomposing Eq. (21) into

$$\begin{aligned} \Delta W_{ik} &= \eta \varsigma_i^\alpha P_k^\alpha = \eta f'(h_i^\alpha)\,[d_i(t) - y_i(t)]\,P_k^\alpha \\ &= \eta f'(h_i^\alpha)\,d_i(t)\,P_k^\alpha - \eta f'(h_i^\alpha)\,y_i(t)\,P_k^\alpha \end{aligned} \tag{25}$$

Similarly, Eq. (23) can be decomposed into

$$\begin{aligned} \Delta V_{kj} &= \eta \varsigma_k^\alpha x_j^\alpha = \eta \left(f'(h_k^\alpha) \sum_i W_{ik} \varsigma_i^\alpha\right) x_j^\alpha \\ &= \eta \left(f'(h_k^\alpha) \sum_i W_{ik} \left(f'(h_i^\alpha)\,[d_i(t) - y_i(t)]\right)\right) x_j^\alpha \\ &= \eta \sum_i \{f'(h_k^\alpha)\, W_{ik} f'(h_i^\alpha)\,[d_i(t) - y_i(t)]\, x_j^\alpha\} \\ &= \eta \sum_i \{f'(h_k^\alpha)\, W_{ik} f'(h_i^\alpha)\, d_i(t)\, x_j^\alpha\} \\ &\quad -\eta \sum_i \{f'(h_k^\alpha)\, W_{ik} f'(h_i^\alpha)\, y_i(t)\, x_j^\alpha\} \end{aligned} \tag{26}$$

If we define some new variables as

$$\Theta_{i,y}^{\alpha} = f'(h_i^{\alpha})\, y_i(t) \tag{27}$$

$$\Theta_{i,d}^{\alpha} = f'(h_i^{\alpha})\, d_i(t) \tag{28}$$

$$\Theta_{k,y}^{\alpha} = f'(h_k^{\alpha}) \sum_i W_{ik} \Theta_{i,y}^{\alpha} \tag{29}$$

$$\Theta_{k,d}^{\alpha} = f'(h_k^{\alpha}) \sum_i W_{ik} \Theta_{i,d}^{\alpha} \tag{30}$$

Then, Eqs. (25) and (26) can be rewritten into

$$\Delta W_{ik} = \eta \Theta_{i,d}^{\alpha} P_k^{\alpha} - \eta \Theta_{i,y}^{\alpha} P_k^{\alpha} \tag{31}$$

and

$$\Delta V_{kj} = \eta \Theta_{k,d}^{\alpha} x_j^{\alpha} - \eta \Theta_{k,y}^{\alpha} x_j^{\alpha} \tag{32}$$

From Eqs. (31) and (32), it is clear that all weights in the network are adapted based on the linear combination of the nonlinear Hebbian and anti-Hebbian rules which are correlation learning as shown by Eq. (8). Therefore, the BP algorithm can be decomposed into a linear combination of correlation learning between the input and the desired signals, the output and the input, the desired signal or the outputs, and some intermediate variables.

It is also interesting to note that all the correlation learnings discussed above are given with the same time indices; namely, τ in Eq. (2) is zero. This point is different from

the weight adjustment rule of recurrent network to be discussed below.

(ii) Real-time recurrent BP algorithms

Conventional BP algorithm is usually used for static neural networks and time delay neural network (TDNN) type networks. Since those networks are limited in capturing the dynamics of the training data, recurrent networks which include feedback connections have to be used. For a recurrent network, the training algorithms could be one of the following: temporal BP [Wan, 1990], Backpropagation through time [Werbos, 1990], recurrent BP [Pineda, 1988], and real-time recurrent learning (RTRL) [Williams and Zipser, 1989]. We will use the RTRL here to illustrate that the correlation concept is contained in this kind of learning algorithms. For other approaches, similar analysis could be applied.

Following the discussion given by Haykin [Haykin, 1994], a recurrent network has arbitrary connections and arbitrary feedback. The weight adjustment of w_{kl}, from the lth neuron to kth neuron, has the form,

$$\Delta w_{kl} = \eta \sum_{j \in \varsigma} e_j(t)\, \pi_{kl}^j(t) \tag{33}$$

where ς is the set of visible neurons which provide externally reachable outputs, $e_j(t) = d_j(t) - y_j(t)$, if $j \in \varsigma$ (and 0), otherwise, and

$$\pi_{kl}^j(t+1) = \varphi'(\upsilon_j(t)) \left[\sum_{i \in B} w_{jl}(t)\, \pi_{kl}^j(t) + \delta_{kj} u_l(t) \right] \tag{34}$$

where B is the set of output neurons, $\varphi(.)$ is the nonlinearity of each neuron, $\upsilon_j(t)$ is the internal activity of neuron j at time n, δ_{kj} is a Kronecker delta function. It should be pointed out that $u_l(t)$ belongs to either B or external inputs, and $\pi_{kl}^j(0) = 0$.

Substituting Eq. (34) into Eq. (33), we have,

$$\Delta w_{kl} = \eta \sum_{j \in \varsigma} e_j(t) \{\varphi'(\upsilon_j(t-1)) \Big[\sum_{i \in B} w_{ji}(t-1) \pi^j_{kl}(t-1) + \delta_{kj} u_l(t-1) \Big]\}$$

$$= \eta \sum_{j \in \varsigma} (\sum_{i \in B} \varphi'(\upsilon_j(t-1)) w_{ji}(t-1) e_j(t) \pi^j_{kl}(t-1) + \tag{35}$$

$$+ \delta_{kj} e_j(t) \varphi'(\upsilon_j(t-1)) u_l(t-1))$$

$$= \eta \sum_{j \in \varsigma} (\sum_{i \in B} \varphi'(\upsilon_j(t-1)) w_{ji}(t-1) (d_j(t) - y_j(t)) \pi^j_{kl}(t-1) +$$

$$+ \delta_{kj} \varphi'(\upsilon_j(t-1)) (d_j(t) - y_j(t)) u_l(t-1))$$

Since the w_{kl} and the activation $\varphi'(\upsilon_j(t-1))$ can be viewed as time-varying coefficients, we have correlation functions for $d_j(t)\pi^j_{kl}(t-1)$ and $y_j(t)\pi^j_{kl}(t-1)$, which are correlation functions at different time lags. The $\varphi'(\upsilon_j(t-1))$, when φ is a sigmoid function, takes the form of

$$\varphi'(\upsilon_j(t-1)) = \upsilon_j(t-1)\,[1 - \upsilon_j(t-1)] \tag{36}$$

It has been seen that the correlation functions contained in the RTRL are correlated for different time indices, which distinguishes the dynamical networks from the static networks. And this is one of the reasons that the recurrent network can capture the dynamics of the signals.

(iii) High-order networks

So far, the correlation functions discussed for the family of supervised learning only included implicitly correlated relationships among the input, the desired signals, and the output. It is possible to build an explicit correlation rule to adjust weights in a supervised neural network. The higher-order neural networks proposed by Lee Giles [Giles et al.,

1987] are good examples of using explicit correlation function in the weight adjustment. It should be pointed out that the higher-order neural networks are static networks, such that the indices i, j, k, and so on represent the relationship among nodes. Of course, it is possible to use similar adaptation rules but with different time indices to adjust the weights.

Without loss of generality, let us use second-order neural networks as an example here. There are two learning approaches for high-order networks, according to [Giles and Maxwell, 1987]. One typical single-layer learning rule is the perceptron rule, which for the second-order network can be expressed as:

$$w_2^{new}(i,j,k) = w_2^{old}(i,j,k) + [d(i) - y(i)]\, x(j)\, x(k) \tag{37}$$

This rule is a supervised rule. Another rule is the outer product rule or Hebbian rule. In this form of learning, the correlation matrix is formed via the equation:

$$w_2(i,j,k) = \sum_{s=1}^{Np} [y^s(i) - y(i)]\, [x^s(j) - x(j)]\, [x^s(k) - x(k)] \tag{38}$$

where Np denotes the number of patterns in the training set. and y and x are the average of y^s and x^s over the training set.

Looking at Eqs. (37) and (38), they are consistent with the definitions of high-order correlation.

2.4 Correlation Rule is a Universal Principle for Learning Systems

Summarizing the results obtained in section 2.3, it is easy to see that the reviewed learning algorithms are a linear combination of correlation functions between the input and the desired signals, the output and the input, the desired signal or the outputs, and some intermediate variables. And it is always true that the output and the desired signals in

the supervised learning are separated; that is, they are never correlated together. This may seems strange due to the nonlinear nature of the network that could mix the signal information together. We will explain this point in this section in detail and show why the second-order correlation is always joined with learning algorithms using gradient information and MSE criterion.

In order to make the separation principle explicit, let us first study the learning processes in a supervised system from the viewpoint of data flow [see Principe, 1996 for details]. It is well known that gradient descent calculations can be implemented as forward (input-output) data flow followed by backward (output-input) data flow. In the forward process, the input data is injected into the network and local activations and the network output are computed from the input. The data flow for the ith processing element (PE) in the forward process is shown in Figure 8a. In the backward process, the output error for each PE is calculated based on the difference between the desired and the output signals. Then this local error is propagated through the dual topology. The data flow of the backward process is given in Figure 8b for comparison with the data flow of the forward process.

The dual network has the same structure as the original network, except it is a linear network which means the summing junctions in Figure 8a become spliting nodes which are linear nodes including summing and multiplied by a coefficients $f'(net)$ in Figure 8b. The error δ_i is from right to left, and the activation is from left to right.

Separation Principle: For a learning system training based on the gradient method and MSE criterion such as BP, RTRL, and so on, the backward process is a linear process such that the error, which is defined as the difference between the desired signal and the output, can be decomposed as a linear combination of functions of the desired signal and output.

With this principle, it is clear why the supervised learnings discussed above can be split into a linear combination of several correlation functions in which the desired signal

and the output nerve appear correlated.

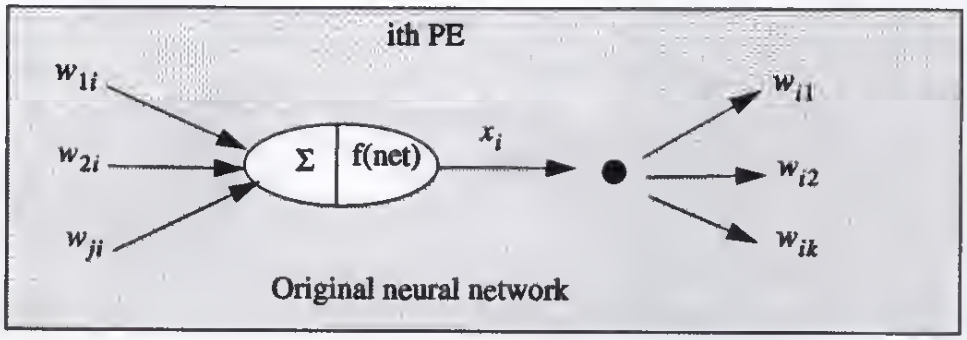


Figure 8a. The data flow of the forward procedure

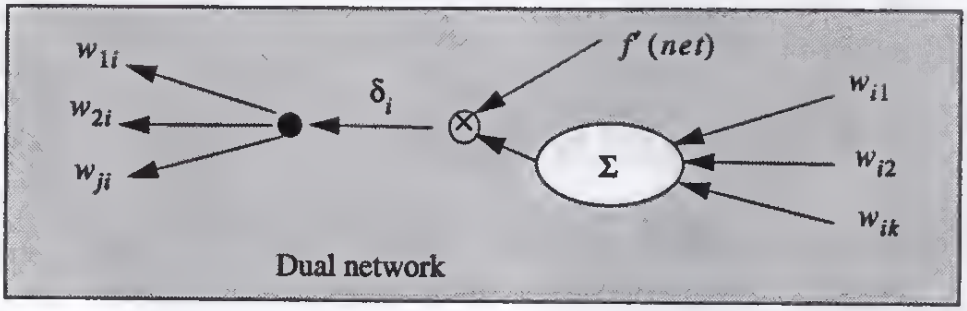


Figure 8b. The data flow of the backward procedure

2.5. Discussions

A generalized learning principle, the correlation learning, for neural networks is outlined based on the analysis of almost all important learning algorithms in neural networks. With the correlation learning theory, it is easy to put the supervised and unsupervised learning in a unified family; that is, the generalized correlation learning rule works for both learning paradigms. Therefore, the problem "what are the principles according to which local synapse changes are organized in order to yield global learning of complex behaviors for the organism?" can be answered at least from one point of view. Moreover,

this unified viewpoint helps us to understand what those signals such as input and desired signals are doing during the learning process. It will be also seen in Chapter 5 that the correlation learning is a bridge between the supervised and unsupervised learning, and the supervised learning will degenerate to unsupervised learning if some correlation terms in correlation learning disappear.

CHAPTER 3

PRINCIPAL COMPONENTS ANALYSIS IN TIME

3.1. Introduction

In chapter 2, we have seen that most of unsupervised learning rules such as Hebbian, competitive, and Principal Components Analysis (PCA) are part of the family of correlation learning. But all of them are static learning rules, that is, the time relationship between input samples was not considered when these algorithms were derived. On the contrary, the supervised paradigm which includes static learning rules such as backpropagation has been extended to backpropagation through time for dynamic networks, for instance, and all have been to be members of correlation learning. Therefore, for completeness it is necessary to study unsupervised learning integrated with the time index. In this chapter, we focus on the PCA learning rule and extend it to a dynamic structure. It should be mentioned that the theory of the PCA in time actually has an analytical solution in signal processing for linear networks. Here, we will focus on an iterative on-line approach which is much more useful in practice.

The direct application of the Hebbian rule in neural networks leads to PCA which is a key technique for features extracting, data compression, and signal representation. Hence, PCA is used extensively in statistics, image processing, communication, etc. PCA learning has been extensively studied by researchers and various PCA networks were proposed [Oja, 1982, 1992; Sanger, 1989; Kung et. al, 1990]. Oja did pioneering work on PCA learning and laid down a foundation for implementing PCA using neural networks;

Sanger extended PCA to the ordered multi-output case and presented examples for important applications; Kung et. al developed a new algorithm called Adaptive Principal Components EXtraction (APEX) for recursive computation of principal components. An attractive feature of the APEX is that if we are given the first j principal component, the algorithm computes the (j+1)th principal components in an iterative manner. Although PCA networks are versatile and useful, they can be used only for static patterns, that is, only suitable for random variables in that the order of patterns is not important for processing. A lot of real-world problems are associated with sequential processing where the order of patterns plays an important role. From the signal processing point of view, a static pattern is a vector of random variables. However, a temporal pattern is a set of random variables dependent upon time, which is called a stochastic process. In order to analyze a random process with PCA networks, it is necessary to study the sequential processing ability of the PCA networks. Only a few papers discuss the topic such as unsupervised learning in time or temporal unsupervised learning [Chappell and Taylor, 1993]. In this chapter, the terms 'temporal processing' and processing or learning in time have the same meaning. For the sake of processing temporal sequences in the unsupervised mode, the approach used in supervised learning was adopted, i.e., transferring the temporal signal into a static pattern with the help of a short term memory structure. Vaz and Principe applied the tapped delay line directly to PCA networks and gave very interesting results for electroencephalogram (EEG) spike detection [Vaz and Principe, 1994], but they did not advance the theoretical analysis of the method. It is, therefore, necessary to pay special attention to unsupervised learning with short term memory components. In supervised learning, a desired signal is given and the network knows what it should learn about the desired signal. However, in unsupervised learning, because the network uses no desired response and recent information may erase previously learned information, the learning result is decided by the learning algorithms as well as the structure of the network. Besides, it will be seen in this chapter that, with the tapped delay, dynamics are added to

the network and its properties are totally changed. It will be shown that this network can access information of the input signal in both time domain and frequency domain. It provides a totally new tool to process nonstationary or quasi-stationary signals such as EEG. Therefore, it is necessary to theoretically analyze what the network is learning.

This chapter is organized as follows: The prerequisite background needed for this chapter is outlined in section 3.2. In section 3.3, we review the static PCA network and propose a structure for Hebbian learning in time and provide a theoretical analysis of the new structure. We point out that with tapped delay line for the input sequence, the PCA networks can do not only principal components decomposition based on autocorrelation, but also capture more information in different domain such as frequency domain. Specifically, for stationary signal with infinite samples this network implements the Fourier transformation, and is optimal in the sense of maximizing the output signal-to-noise ratio; filter bank and eigenfilter interpretations are given in section 3.4. Some problems associated with nonstationary processing will be discussed in section 3.5 and a multi-resolution network will be proposed there. Section 3.6 presents simulation results for the short time-frequency decomposition of an EEG signal, signal reconstruction using a multi-resolution network, and an application of the PCA in time in adaptive filtering. Discussion and remarks are stated in section 3.7.

PCA in time is a very rich topic, and a complete analysis for nonstationary input signal case is very difficult due to the time-varying property of the signal. In this chapter, however, we will highlight some properties of the PCA in time network and also present relationship between PCA in time and some conventional digital signal processing algorithms.

3.2 Some Basic Concepts and Definitions

In this section, we review the basic knowledge of random variables and random pro-

cesses, which are necessary to understand the relation between static and temporal patterns from the mathematical point of view and also serves as a basis of the analysis to be presented in this chapter.

3.2.1 Random Variables

Definition of random process [Papoulis, 1991]: A random variable x is a process of assigning a number $x(\zeta)$ to every outcome ζ of an experiment.

Definition of random vector [Papoulis, 1991]: A random vector is a vector

$$X = [x_1, ..., x_n] \tag{39}$$

whose components x_i, i=1,...,n, are random variables.

Definition of correlation matrix [Papoulis, 1991]: The correlation matrix is defined as:

$$R_n = \begin{bmatrix} R_{11} \cdots R_{1n} \\ \cdots \quad \cdots \quad \cdots \\ R_{n1} \cdots R_{nn} \end{bmatrix} \tag{40}$$

where

$$R_{ij} = E[x_i x^*_j] \tag{41}$$

and * is the conjugate transpose operator.

Properties:

(1) The matrix R_n is nonnegative definite.

(2)

$$R_{ij} = R^*_{ji} \tag{42}$$

Clearly,

$$R_n = E[X^T X^*] \tag{43}$$

where T is transpose operator.

Definition of covariance matrix [Papoulis, 1991]: The covariance matrix is defined as

$$C_n = R_n - \begin{bmatrix} \eta_1\eta_1 & \cdots & \eta_1\eta_n \\ \cdots & \cdots & \cdots \\ \eta_n\eta_1 & \cdots & \eta_n\eta_n \end{bmatrix} \tag{44}$$

where $\eta_i = E[x_i]$, i=1,...,n, is the mean values of random variables. When the mean of the random variable is zero, covariance matrix and autocorrelation matrix are the same.

Note that there is no time index contained in the variables defined above.

3.2.2 Random Process

Definition of random process [Papoulis, 1991]: A random process x(t) is a process that depends both on the outcome of an experiment and on time t. The time index could be either continuous or discrete, but it is always assumed that time is discrete in this chapter if it is not specified.

The following points are necessary to understand the difference and connection between a random process and a random variable:

(1) It is a family of functions $x(t, \zeta)$. In this interpretation, t and ζ are variables.

(2)It is a single time function. In this case, t is a variable and ζ is fixed.

(3) If t is fixed and ζ is a variable, then x(t) is a random variable equal to the state of the given process at time t.

(4) If t and ζ are fixed, then x(t) is a number.

Definition of autocorrelation [Papoulis, 1991]: The autocorrelation $R(t_1, t_2)$ of x(t) is the expected value of the product $x(t_1)\,x(t_2)$:

$$R(t_1, t_2) = E\{x(t_1)\,x(t_2)\} \tag{45}$$

Notice that $R(t_1, t_2)$ is a matrix of time.

Properties:

(1) The matrix $R(t_1, t_2)$ of x(t) is positive definite.

(2)

$$R(t_2, t_1) = R(t_1, t_2)^* \tag{46}$$

Definition of autocovariance [Papoulis, 1991]: The autocovariance $C(t_1, t_2)$ of a process x(t) is defined as:

$$C(t_1, t_2) = R(t_1, t_2) - \eta(t_1)\,\eta^*(t_2) \tag{47}$$

For zero mean random process x(t) or zero mean random vector X, the autocorrelation function is equal to the covariance (autocovariance) function.

If the random process is assumed stationary, we have

$$R(t_1, t_2) = R(t_2 - t_1) = R(\tau) \tag{48}$$

where $\tau = t_2 - t_1$.

One aspect of the connection between a random process and a random vector could be clarified from the following point: Sampling a realization of a random process with fixed time step form an ordered random vector. Actually, this is the strategy used in temporal processing to transfer a temporal pattern into a static one, and can be realized easily with the help of a delay line.

From the previous definitions and notes, we can find that both random process x(t) and random vector X are vectors and a random vector is a special case of a random process when the time t is fixed. Additionally, the autocorrelation matrices defined for the random process and the random vector have the same form except that their elements are decided by both the random variables and time for $R(t_1, t_2)$ and by the random variables only for R_n.

3.3. PCA Learning In Time

3.3.1 Static PCA Network

A network for PCA was given in Figure 1, and we reproduce it here in Figure 9 for convenience. The input is denoted by $X = [x_1, \ldots, x_p]$ and output denoted by $Y = [y_1, \ldots, y_m]$.

It is assumed that $E[X] = 0$ and the weight vector is a unitary vector. Under these conditions, it is not difficult to show [Haykin, 1994] that the weight $W = [w_1, \ldots, w_m]$ for normalized Hebbian rule (Oja rule), where w_k is defined as the connection between the input vector and the kth output neuron $w_k = [w_{k1}, \ldots, w_{kp}]^T$, is related to the autocorrelation matrix R_p of the input signal in the form of

$$R_p W = W\Lambda \tag{49}$$

where Λ is a diagonal matrix defined by the eigenvalues of matrix R_p, namely,

$$\Lambda = diag[\lambda_1, ..., \lambda_m] \tag{50}$$

and $\lambda_1, ..., \lambda_m$ are the largest eigenvalues of the R_p in decreasing order. Therefore, the weight matrix W is the normalized eigenvector of the autocorrelation matrix R_p.

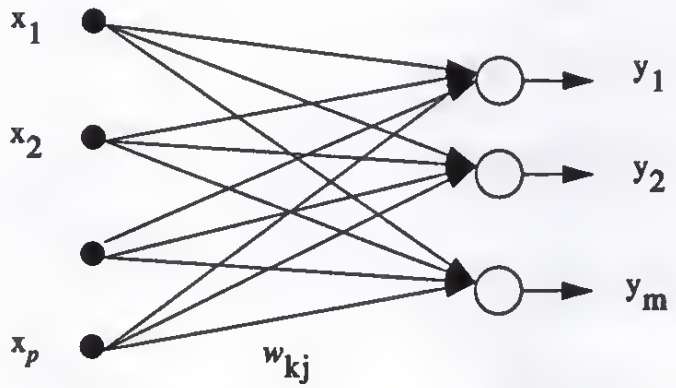


Figure 9. A linear network for static PCA

The static PCA learning includes two steps. One is search for the eigenvector of R_p, and the second one is the projection of the original signal to the eigenvectors sub-space. There are two equations---the analysis and synthesis equations, to describe the two steps. The analysis equation has the form of:

$$X = \sum_i y_i w_i \tag{51}$$

where i is index bases for signal representation and may go to infinity for the complete representation of the input signal X. And the synthesis equation is:

$$Y = WX = \sum_{j=0}^{m-1} x_j w_j \tag{52}$$

where m is the dimension of the output layer.

Actually, the analysis and synthesis equations given in Eqs. (51) and (52) are validate for time signal and produce other linear representations of signal such as Fourier transformation and wavelets. But for the time signal the variables used above are time-varying and should be denoted as $X = [x_1(t), \ldots, x_m(t)]$. The major difference for various signal representation algorithms is that signal subspace is constituted in different ways. For example, the Fourier bases are complex exponential for complex signal, the PCA bases are decided based the variance of the signal, and various wavelet bases are available if a scale which controls the width of the window is used.

In the PCA representation, the bases for analysis equation are given based on Eq. (49). And it is straightforward to show the variances of each output unit $\mathrm{var}(w_k) = \lambda_k$. But it should be pointed out that the PCA has a lot more structure than the Fourier transformation and wavelets since the basis of the PCA sub-space are ordered based on the variance or energy of the projected signals, which is critical for optimal signal representation using finite number of bases. With the help of Hebbian rule, PCA can be implemented with an on-line version algorithm. It is well known that an on-line implementation is usually the most important aspect for real-world problems.

Summarizing the discussion above, we have:

(1) The eigenvectors of the autocorrelation matrix R_p pertaining to the zero mean data vector X define the unit vector w_k, represents the principal directions along which the variance probes $\mathrm{var}(w_k)$ have their extreme values.

(2) The analysis and synthesis equations constitute the two steps in PCA. And the ordered eigenvectors provide the optimal representation (in the mean square sense) of a signal with finite number of bases.

(3) On-line adaptive learning rule, like the Hebbian rule, provides a very powerful tool for PCA.

3.3.2 PCA for Random Process.

We have seen that with the network given in Figure 9, the Oja rule can process only static signals, that is, random variables. In order to design a PCA network to process time sequences using Oja rule, it is necessary to propose a novel network structure to access the time information embedded in a random process.

We adopt the idea of processing time sequences in supervised networks such as TDNN and Gamma network, and use a memory structure (tapped delay line here) in the network as shown in Figure 10.

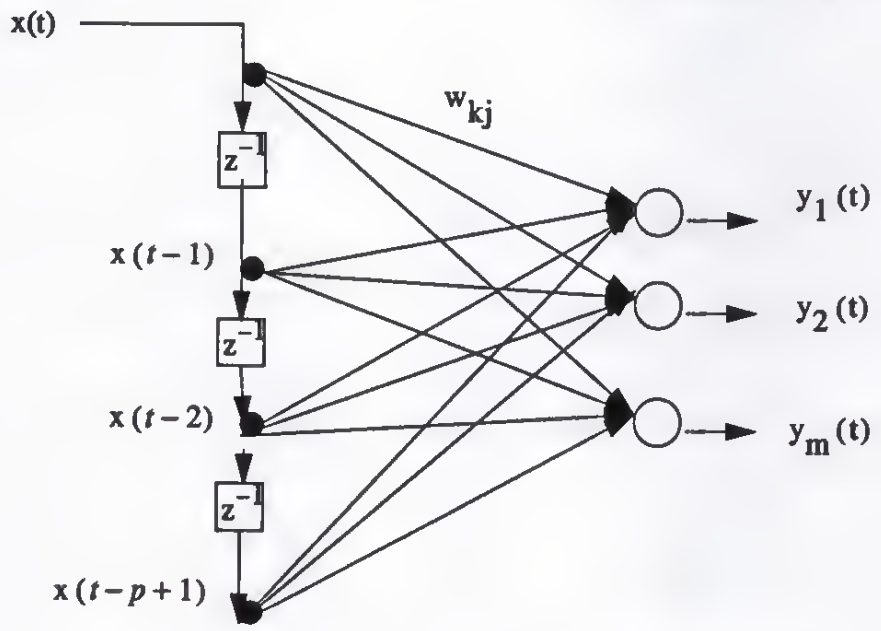


Figure 10. A linear network with a tapped delay line in the input layer

In this network, the input signal is shifted into the tapped delay line one by one, and the weights are still adjusted by the Oja rule. The output signals are obviously a function

of time index t. Actually, this structure can be viewed as a filter bank trained with modified eigenfilter algorithm as discussed in section 3.4. Moreover, it will be clarified in Chapter 5 that the Hebbian rule is a building block for the LMS algorithm, but here, we will focus on the Oja rule.

For the network given in Figure 10, we adopt the mathematical analysis for time sequence processing using PCA [van Trees, 1968; Fukunaga, 1990] to illustrate its properties.

A random process x(t), defined in a time interval [0,T], can be expressed as a linear combination of basis functions [van Trees, 1968; Fukunaga, 1990], which is actually the so-called synthesis equation:

$$x(t) = \sum_{i=0}^{\infty} y_i(t)\, w_i(t) \qquad (0 \le t \le T) \tag{53}$$

where the basis functions $w_i(t)$ are deterministic time functions and the coefficients y_i are random variables. An infinite number of $w_i(t)$ is required in order to form a complete set. The orthonormal condition of $w_i(t)$ is given by

$$\int_0^T w_i(t)\, w_j^*(t) = \delta_{ij} \tag{54}$$

where * is the complex conjugate operator, and δ is delta function.

For a zero mean random process x(t), the autocorrelation function is defined as:

$$R(t, \nu) = E[x(t)\, x(\nu)^T] \tag{55}$$

If the $w_i(t)$'s are the eigenfunctions of $R(t, \nu)$, they must satisfy the following integral equation:

$$\int_o^T R(t, \nu) w_i(t) \, d\nu = \lambda_i w_i(t) \qquad (i = 1, 2, \ldots) \tag{56}$$

where λ_i's are the eigenvalues of $R(t, \nu)$.

Eqs. (53) and (56) are the core for analysis equations for temporal PCA, which are extensions of the equations given in Eqs. (51) and (49) for static PCA.

Suppose we take n time-sampled values of these time functions and convert them to vectors as:

$$X = [x(t_1) \ldots x(t_n)]^T \tag{57}$$

$$\phi_i = [w_i(t_1) \ldots w_i(t_n)] \tag{58}$$

where each time-sampled value of x(t), $x(t_i)$, is a random variable. Then, for example, Eqs.(54) and (56) can be rewritten as follows:

$$\sum_{k=1}^{n} w_i(t_k) w_j^*(t_k) = \phi_i^T \phi_j^* = \delta_{ij} \tag{59}$$

$$\sum_{k=1}^{n} R(t_l, t_k) w_i(t_k) = \lambda_i w_i(t_l) \quad i, l = 1, 2, \ldots, n \tag{60}$$

Eq. (60) can be written in a matrix form to define the eigenvalues and eigenvectors as

$$S\phi_i = \lambda_i \phi_i \qquad (i = 1, 2, \ldots) \tag{61}$$

where S is

$$S = \begin{bmatrix} R(t_1, t_1) & \dots & R(t_1, t_n) \\ \dots & \dots & \dots \\ R(t_n, t_1) & \dots & R(t_n, t_n) \end{bmatrix} = \begin{bmatrix} E[x(t_1)x(t_1)^T] & \dots & E[x(t_1)x(t_n)^T] \\ \dots & \dots & \dots \\ E[x(t_n)x(t_1)^T] & \dots & E[x(t_n)x(t_n)^T] \end{bmatrix} \quad (62)$$

Since S is an n-by-n matrix, we can obtain only n eigenvalues and eigenvectors instead of an infinite number.

The difficulty in the temporal PCA is that we have to solve the matrix Eq. (60) in order to obtain eigenvalues and eigenfunctions. But we should stress that [van Trees, 1968]:

(1) There exists at least one square-integrable function $\phi_i(t)$ and an associative real number $\lambda_i \neq 0$ that satisfy Eq. (60).

(2) We can always normalize the eigenfunctions.

(3) The eigenfunctions corresponding to different eigenvalues are orthogonal.

Recently, the stability analysis of the PCA algorithm has been developed [Elfadel, 1995; Plumbley, 1995]. A Lyapunov function was built for the dynamics of the PCA learning, and the globally convergent analysis was given to show that the signal subspace spanned by a set of orthonormal weight vectors will asymptotically converge to the principal subspace from almost everywhere if the weight matrix W remains finite and full rank [Elfadel, 1995; Plumbley, 1995].

Therefore, for a correlation function with different eigenvalues, there always exists a set of orthonormal eigenfunctions satisfying Eq. (60). Furthermore, these orthogonal basis functions are ordered based on the projected signals' energy, which provides an optimal solution for signal compression and information extraction in the Mean Square Error (MSE) sense. The adaptation ability of the Hebbian rule is a very powerful tool because the basis functions are changed adaptively with the second-order statistical information of the input signal. For some special signals such as stationary signal, the properties of the

temporal PCA will be illustrated in the subsequence section.

3.3.3 Principal Components Analysis for Stationary Random Sequence

Although the solution of Eq. (60) is not easy to find generally, for some special cases, explicit solutions can be found. The most fundamental case is the stationary random process.

Assuming that x(t) is a stationary process, we have

$$R(t, \nu) = R(t-\nu) \tag{63}$$

And Eq. (56) becomes [Fukunaga, 1990],

$$\int_{-\frac{T}{2}}^{\frac{T}{2}} R(t-\nu)\, w_i(\nu)\, d\nu = \lambda_i w_i(t) \qquad \begin{matrix} (i = 1, 2, \ldots) \\ (\frac{T}{2} \le t \le \frac{T}{2}) \end{matrix} \tag{64}$$

where the time region is shifted from [0,T] to [-T/2, T/2] for convenience.

Let us extend T to infinite. Then, we have

$$\int_{-\infty}^{\infty} R(t-\nu)\, w_i(\nu)\, d\nu = \lambda_i w_i(t) \qquad \begin{matrix} (i = 1, 2, \ldots) \\ (-\infty \le t \le \infty) \end{matrix} \tag{65}$$

Since Eq. (65) is the convolution integral of R(t) with $w_i(t)$, the Fourier transform of this equation becomes

$$\varsigma(j\omega)\, \Xi_i(j\omega) = \lambda_i \Xi_i(j\omega) \tag{66}$$

where $\varsigma(j\omega)$ and $\Xi_i(j\omega)$ are the Fourier transforms of R(t) and $w_i(t_1)$, respectively. Particularly, $\varsigma(j\omega)$, the Fourier transform of the autocorrelation function of a random process, is known as the power spectrum of the random process x(t).

In order to solve Eq. (66), let us assume that $\varsigma(j\omega)$ is nowhere flat, the solution must be an impulsive function as [Fukunaga, 1990]

$$\Xi_i(j\omega) = \delta(\omega - \omega_i) \tag{67}$$

which corresponds, in time domain, to

$$w_i(t) = e^{j\omega_i t} \tag{68}$$

then,

$$\varsigma(j\omega) = \lambda_i \tag{69}$$

and the PCA can be reformulated as

$$x(t) = \sum_{i=-\infty}^{\infty} y_i e^{j\omega_i t} \tag{70}$$

With ω_i changing continuously from $-\infty$ to ∞, the summation of Eq. (70) is replaced by an integration as

$$x(t) = \frac{1}{2\pi} \int_{-\infty}^{\infty} y(\omega) e^{j\omega_i t} d\omega \tag{71}$$

where $y_i = y(\omega_i)\Delta\omega/2\pi$. Notice that x(t) is the inverse Fourier transform. Thus, when a random process is stationary, the basis functions of the PCA expansion become the

complex exponential, $e^{j\omega_i t}$, and the coefficient, y_i becomes the Fourier transform of x(t). Furthermore, since the correlation matrix is positive, the eigenfunctions form a complete orthonormal set [van Trees, 1968]. Hence, with infinite samples, namely, with infinite taps in the tapped delay line, the reconstruction signal is a Fourier transformation with respect to the weights.

It should be pointed out that the above analysis was derived under the condition of complex basis functions and complex coefficients. It is straightforward to modify it for real basis functions and real coefficients as follows [Fukunaga, 1990]. Since the ith and -ith basis functions, $e^{j\omega_i t}$, $e^{-j\omega_i t}$, are complex conjugate, the coefficients, y_i and y_{-i}, are also conjugate for real x(t). Therefore, combining these two terms, Eq. (71) becomes

$$x(t) = y_0 + \sum_{i=1}^{\infty} 2|y_i| \cos(\omega_i t + \angle y_i) \tag{72}$$

where $|y_i|$ and $\angle y_i$ are the magnitude and the angle of a complex variable y_i. Hence, the eigenvectors become $\cos(\omega_i t + \angle y_i)$, which will be also shown experimentally in section 3.6.

The bases of this inverse Fourier transform are also the eigenvectors of the autocorrelation function of the input data. Thus, this decomposition is not only a Fourier decomposition, but also a PCA decomposition in the sense that the basis functions are ordered and represent the principal components of the input sequence.

The conclusions above are only valid for infinite samples of a random process. For a finite stationary random sequence, the relation between eigenvectors and the corresponding eigenvalues are given by the following theorem:

Theorem: Let x(t), t=-T,...., T, is a stationary random sequence. The eigenvectors and eigenvalues of the PCA of x(t) are given approximately by [Brillinger, 1980]:

$$2(2T+1)^{1/2}[\cos(2\pi st/(2T+1)); t = \overline{-T}, \ldots, T] \tag{73}$$

and

$$\sum_{t=-2T}^{2T} 2R(t)\cos(2\pi st/(2T+1)) \tag{74}$$

respectively s=0,...,T. The principal components are therefore approximately

$$2(2T+1)^{1/2}\sum_{t=-T}^{T}\cos(2\pi st/(2T+1))\,x(t) \qquad s=0,\ldots,T \tag{75}$$

Obviously, each principal component, as given by Eq. (75), includes contributions from different frequencies. This is a very important observation which shows that the signal generated by each principal component of the PCA algorithm can be viewed as an output of a bandpass filter, and the PCA network, therefore, is a filter bank with different pass bands. This is a different point from the Fourier transformation in which each component represents a single frequency.

Summarizing the above, we see that the output of an infinitely large network is the Fourier transform of the input signal and each axis represents different principal components or power of the signal for frequency ranging from $(0, \infty]$ (note that there is no negative frequency and Eq. (71) can be rewritten into a summation only with positive frequency [Fukunaga, 1990]). Hence, the PCA for a random process presents the feature of a signal in the frequency domain. For the finite data samples, it is still a Fourier transformation with modified basis functions as given in Eq. (74).

It should be emphasized that although the above discussions were given from the signal processing point of view, mathematical foundation of the above argument was based on Theorem 3.3.1 which says the PCA basis is asymptotically convergent to the Fourier basis even with finite data. Of course, the larger number of the samples, the better

the approximation.

3.3.4 On-Line Implementation of Temporal PCA

As mentioned previously, PCA learning can be implemented by Oja's rule (Eq. (9)). But for the temporal PCA network, the time index should be included in the Oja's rule. Hence, we propose to extend the Oja's rule for temporal PCA as

$$w_{kj}(t+1) = w_{kj}(t) + \eta y_k(t) x_j(t) - \eta y_k(t) y_k(t) w_{kj}(t) \tag{76}$$

where t is the discrete-time index of the input signal.

Two problems are associated with this learning rule. The first one is its stability, and the second one is to find which value the learning rule converges.

The dynamical equation given by Eq. (76) can be depicted in Figure 11.

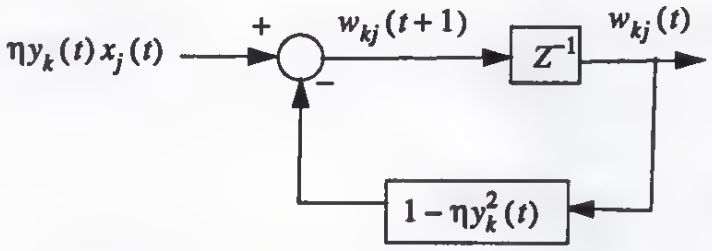


Figure 11. The block diagram of the learning rule in Eq. (76)

The transfer function of the system given in Figure 11 is:

$$G(z) = \frac{1}{z + (\eta y_k^2(t) - 1)} \tag{77}$$

The stability condition for G(z) is:

$$-1 \le \eta y_k^2(t) - 1 \le 1 \tag{78}$$

$$0 \le \eta \le 2/y_k^2(t) \tag{79}$$

For finite energy input signal x(t), the output $y_k(t)$ is also finite because of the input-output relationship $y(t) = wx(t)$ and the constrain $||w|| = 1$, where $y = [y_1, \ldots, y_m]$ is the output vector and $||\ ||$ denotes the Euclidean norm. So, if the step size is selected as a nonnegative number and is not larger than $2/y_k^2(t)$, the learning rule is stable. Just like in the LMS algorithm, we usually don't have lots of knowledge about the input and the output signal, so the step size should be selected as a small number [Windrow and Stearns, 1986]. Notice that the constraint finite input is also true in practice.

The second problem can be understood from the following points: Since a tapped delay line is added at the input, the actual input to the network only contains a fixed number of samples of the whole sequence at each time instant. Then, the correlation function which is computed from the actual input is effectively an estimation of the short-time autocorrelation function for stationary signal which has the form of [Rabiner and Schafer, 1978]

$$\hat{R}(t, \nu) = E[x(t)\, g(k-t)\, (x(\nu)\, g(t-\nu))^T] \tag{80}$$

where g(.) represents a moving window function which is a tapped delay line here, and v is also a time index.

Eq. (80) can be interpreted as follows: First, a segment of the signal is selected by multiplying the window with the original signal; then the correlation function is applied to

the windowed segment of the signal; namely, for each time instant, the network calculates the local correlation function defined by Eq. (80). But the samples of the input signal are shifted one by one, and the learning is repeated not only for each sample, but for each iteration also. Hence, it is critical to know whether the local correlation function will converge to the true correlation function which is computed using the whole samples. Fortunately, this has been shown by Picinbono [Picinbono, 1993]. Picinbono pointed out: when the number of samples is much larger than the number of taps, namely, $n \gg p$, and the tapped delay line is long enough, the autocorrelation matrix in Eq. (80) is a correct estimation of the autocorrelation matrix of x(n). $n \gg p$ is easy to satisfy in an adaptive system since it always takes a large number of iterations of samples in order for the system to reach the stationary point. Hence, we can say that the PCA network with a tapped delay is working with the time correlation instead of the sample autocorrelation. Therefore, the conclusion is that for a stationary signal the network given in Figure 10 is able to estimate the principal components of a stationary sequence if the tapped delay line is long enough.

All experiments given in section 3.6 will employ this Oja's rule to train the temporal PCA network.

For a stationary input signal, the structure given in Figure 10 with enough number of delay units is also an optimal system in the sense of signal noise ratio if we study it from the eigenfilter point of view, which will be explained in the next section.

3.4 Eigenfilter and Filter Bank Interpretation

Consider a linear FIR filter shown in Figure 12 whose impulse response is {w}. The sequence x(t) applied to the filter input consists of a useful signal component s(t) plus an additive noise component n(t). The signal s(t) is drawn from a wide-sense stationary stochastic process of zero mean and correlation matrix R. The noise component n(t) is white with a constant power spectrum density of variance σ^2. The filter output is denoted by

y(t). The situation described herein is depicted in Figure 12 [Haykin, 1991; Makhoul, 1981].

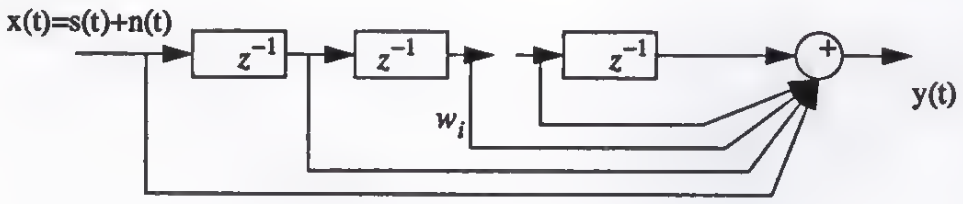


Figure 12. The FIR eigenfilter

It can be seen that the filter in Figure 12 is one component of the filter bank (Figure 13) description of the network shown in Figure 10. Therefore, this FIR eigenfilter could be used as a building block for the PCA in time network of Figure 10.

We will show that one branch of the PCA network is optimal in the sense of maximizing the output signal-to-noise ratio through the analysis of the eigenfilter given below.

Since the filter of Figure 12 is a linear filter, the superposition principle can be applied. Hence, we may consider the effects of signal and noise separately. Let P_0 denote the average power of the signal component of the filter output y(t). We may show that [Haykin, 1991]

$$P_0 = w^T R w \tag{81}$$

Similarly, the effect of noise alone, the average power is

$$N_0 = \sigma^2 w^T w \tag{82}$$

Let $(SNR)_0$ denote the output signal-to-noise ratio. We may thus write

$$(SNR)_0 = \frac{P_0}{N_0} = \frac{w^T R w}{\sigma^2 w^T w} \tag{83}$$

The optimally may be stated as follows: Determine the coefficient vector w of the FIR filter so as to maximize the output signal-to-noise ratio $(SNR)_0$ subject to the constraint

$$w^T w = 1 \tag{84}$$

Eq. (82) shows that except for the scaling factor $1/\sigma^2$, the output signal-to-noise ratio $(SNR)_0$ is equal to the Raleigh quotient of the coefficient vector w of the FIR filter. We see therefore, that the optimal filtering problem (maximizing SNR), as stated herein, may be viewed as an eigenvalue problem. Indeed, the solution to the problem follows from the minimax theorem [Haykin, 1991]. Specially, using the special form of the minimax theorem, we may state the following:

(1). The maximum value of the output signal-to-noise ratio is given by

$$(SNR)_0 = \frac{\lambda_{max}}{\sigma^2} \tag{85}$$

where λ_{max} is the largest eigenvalue of the autocorrelation matrix R.

(2). The coefficient vector, occurring when the optimal FIR filter that yields the maximum $(SNR)_0$ of Eq. (85), is defined by

$$w_o = w_{max} \tag{86}$$

where w_{max} is the eigenvector associated with the largest eigenvalue λ_{max} of the autocorrelation matrix R.

An FIR filter whose impulse response has coefficients equal to the elements of an eigenvector is called an eigenfilter [Makhoul, 1981]. Accordingly, we may state that the maximum eigenfilter is an optimal filter.

Since in the PCA network, the weight vector is the eigenvector of the autocorrelation matrix, and it satisfies the constraint of Eq. (84), each building block is an eigenfilter. Hence, the network given in Figure 10 for stationary input signal consists of a bank of the m largest eigenfilters. And it is, therefore, optimal in term of output signal-to-noise ratio.

The temporal PCA network given in Figure 10 can be depicted as a filter bank structure as given in Figure 13.

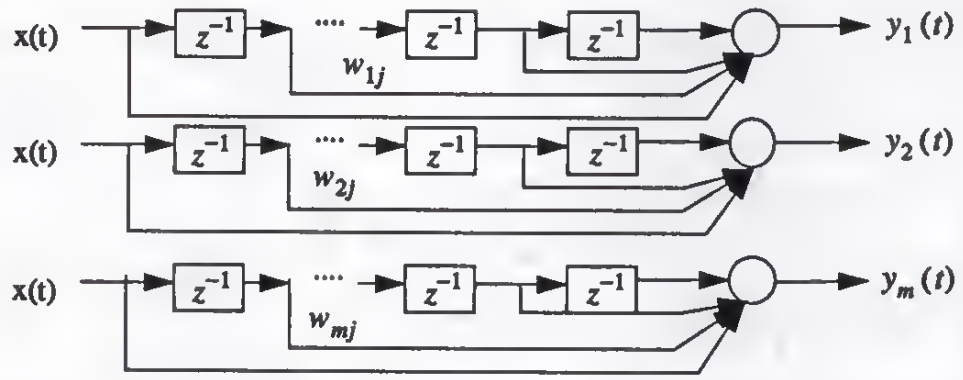


Figure 13. Filter bank interpretation of PCA in time network

In the filter bank, all filters in different channels share the same input signal, and the weights in each channel are adjusted by Oja's rule. And the adaptive ability of the Oja's rule is very powerful for tracking the variation of the signal, temporal PCA network may be used as a signal processing front end for nonstationary or locally stationary signals such as EEG. We will discuss some particular problems for processing nonstationary in the next section.

3.5 Nonstationary Signal Processing Using the Temporal PCA Networks

We will study two different approaches for nonstationary signal processing in this section. First, short time-frequency analysis will be discussed. Then, a multi-resolution network based on the temporal PCA will be proposed. In addition, we will discuss the

selection of the step size and the window, which are critical in nonstationary signal processing.

3.5.1 Short Time-Frequency Analysis Using the Temporal PCA Network

It has been seen that the PCA learning for a stationary random signal approximates the Fourier transform even for finite samples, but for nonstationary signal those results are not necessarily true. Since nonstationary signals such as speech and EEG are very important in many areas, it is very useful to extend the PCA to process them. However, it is well known that strict mathematical analysis is impossible for general nonstationary signals, so some assumptions on the nonstationary signals are needed. For speech signals, the underlying assumption is that the properties of the signals change relatively slowly with time [Rabiner and Schafer, 1978], which means the statistics of the signal change very slowly; namely, the signal in a short duration can be considered approximately stationary. This assumption leads to a variety of 'short-term' processing methods in which short term segments of the signal are isolated and processed as if they were segments from a sustained sound with fixed properties. Often these short segments, which are sometimes called analysis frames, overlap one another. The result of the processing on each frame may be either a single number, or a set of numbers. Therefore, such processing procedures produce a new time-dependents sequence which can serve as a representation of the signal. The new time-dependent sequence contains both time and frequency information. Hence, it is also called short time-frequency analysis. We will give an example in next section to illustrate the short time-frequency analysis using the temporal PCA learning.

3.5.2 A Multi-Resolution Network Based on the Temporal PCA Learning

Since all eigenfilters in the filter bank given in Figure 13 share the same size win-

dow, the resolution of each filter are the same due to uncertainty principle [Gabor, 1946]:

$$\nabla t \nabla f \geq \frac{1}{2} \tag{87}$$

where ∇t is the effective time duration and ∇f is the effective bandwidth of an arbitrary waveform. The inequality of Eq. (87) states that shape localization in time and in frequency are mutually exclusive. Gaussian shaped signals satisfy Eq. (87) with equality.

In order to have a multi-resolution system, the window size for each eigenfilter should be different, based on the uncertainty principle. We propose, therefore, a network shown in Figure 14 for multi-resolution analysis.

The difference between this network and the network given in Figure 10 is that the output units, except the first one $y_1(t)$, are partially connected to the tapped delay line, such that the window or the number of units in the delay line that each eigenfilter employs is different. Since the outputs of the temporal PCA learning are ordered based on the variance and most energy of practical signals is concentrating on the lower frequency area, the window size for the output units may be decreased. Although the multi-resolution network may not converge to the principal components of the input signal, experimental results presented in next section will show that it works better than the temporal PCA network.

It is straightforward to show that the multi-resolution network also obeys the analysis and synthesis equations given by Eqs. (51) and (52) if some of the weights are set zero. Since wavelet representation shares the same analysis and synthesis equations as pointed out in section 3.3.3, we will compare the proposed multi-resolution network with a wavelet representation in the experiments in section 3.6.2 for signal reconstruction task. We should stress that the multi-resolution network is not a wavelet network since there is not mother wavelet and scale associated it, but we will show that the multi-resolution network can outperform a wavelet in signal reconstruction in section 3.6.2.

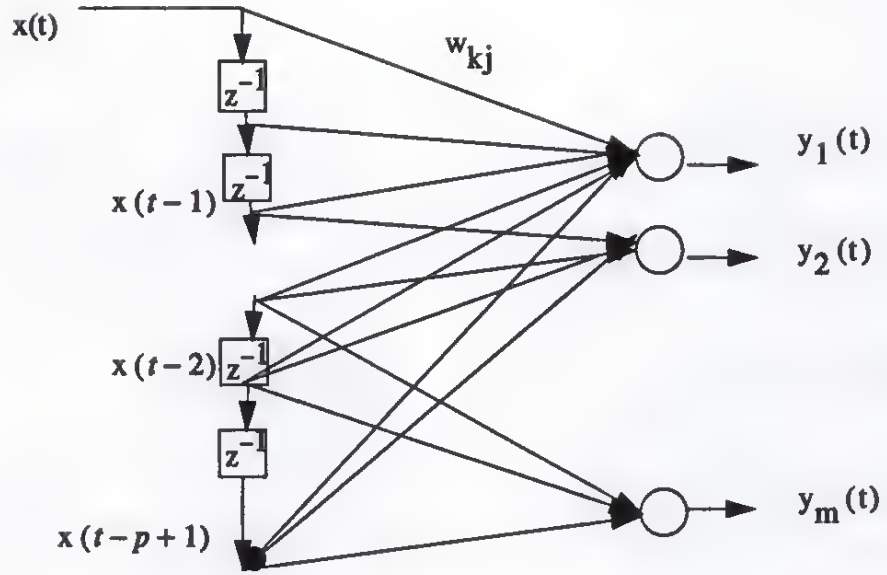


Figure 14. A multi-resolution system based on the temporal PCA network

3.5.3 Selection of Window Size and Learning Step Size

It is very important to select the window size and learning step size in an iterative algorithm, mainly if the input signal is a nonstationary or locally stationary signal. In this section, we propose several practical approaches based on our simulation on the EEG signals.

The selection of window size for the task of locally stationary signal (EEG) reconstruction usually follows the requirements that the window size should be longer than a locally stationary period. At the same time, the window size may be chosen as small as possible in order to avoid blurring the signal in the frequency domain. Another consideration is the computation complexity. An increase of window size implies more weights and the computation complexity is increased. Therefore, the trade-off for window size is important in an iterative algorithm. For the multi-resolution network given in Figure 14, the window size for each eigenfilter is selected in decreasing order because of the decreas-

ing of variances of the projected signals in the output units and the main power of the signal is in the lower frequency region.

In a nonstationary environment, the selection of step size for a iterative learning algorithm is different from the case of stationary input signal, since for the latter the goal is to keep the algorithm convergent, and for the former tracking the statistical variations of the nonstationary signal is the requirement [Haykin, 1996]. So, the step size cannot be selected as a decreasing function of the iteration number. The step size is usually selected as a larger constant to track the input signal. And it is signal dependent.

More illustration of the window and step size selection will be given in the next section with the examples presented there.

3.6 Examples

In this section, two examples and one application of the temporal PCA networks are given. In the first example, we provide some simulation results of the time-frequency representation by temporal PCA learning. In the second example, simulation results using the multi-resolution network will be presented. Then, we apply the PCA in time learning to a transform domain adaptive filter in order to do on-line, orthogonal, time domain filtering. These results show some important properties such as time-frequency and on-line optimal orthogonal transformation.

3.6.1. Experimental Results for Time-Frequency Analysis Using PCA In Time Learning

In this simulation, the network used here is given in Figure 10, which contains a tapped delay line and on-line Oja training algorithm. A segment of an electroencephalogram (EEG) signal sampled at 250Hz is used as an input signal. It is well known that EEG signal is a locally-stationary signal generated by the brain. Figure 15 shows a locally sta-

tionary EEG signal. After 20 iterations, we stopped the training and obtained the decomposed signals. The decomposition of this EEG obtained with ten taps delay line, ten output units, and learning rate 0.012 is given in Figures 16 and 17. The reconstructed signal using the first four output signals is given in Figure 18. The reconstruction method used here is just simply adding all the signals given in the output units of the network in Figure 10 together, since it has been shown that this gives a time domain signal [Shynk, 1992]. Although the reconstruction becomes better if more decomposed signals are used, the main advantage of the PCA learning loses its importance. Hence, a trade-off between the accuracy requirement for reconstruction and the computation complexity and memory space should be always considered whenever PCA-type algorithms are applied. It is easy to see that the reconstructed EEG signal given in Figure 18 is very similar to the original EEG signal presented in Figure 10. The Mean square error between these two signals is 2.43×10^{-4}. The error signal between the original and reconstructed signals is shown in Figure 19. It is seen that errors become larger for samples with lager amplitude.

Two important facts are the window size and the learning rate, which must be selected carefully. We also tried different window size and step size, and the results are presented here. With large window size with 30 delay units and step size 0.02, the decomposed signals are shown in Figures 20 and 21. The reconstructed signal is given in Figure 22. The MSE is 0.0090, and the error signal looks similar to curve in Figure 19. The MSE is larger than the MSE with 10 delay units.

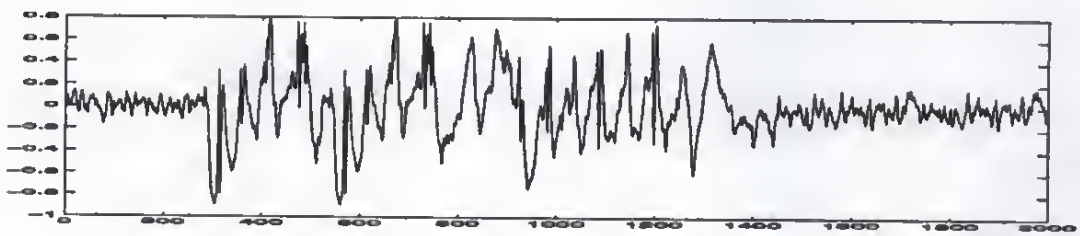


Figure 15. A EEG signal

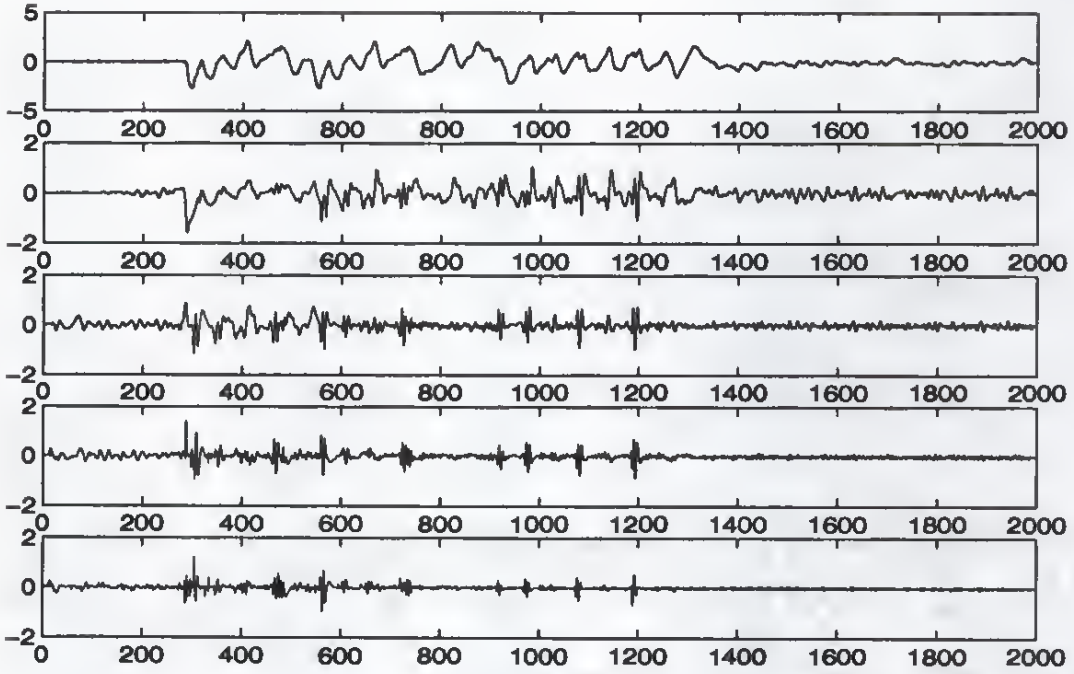


Figure 16. The first five PCA decompositions of the EEG signal with 10 delay units

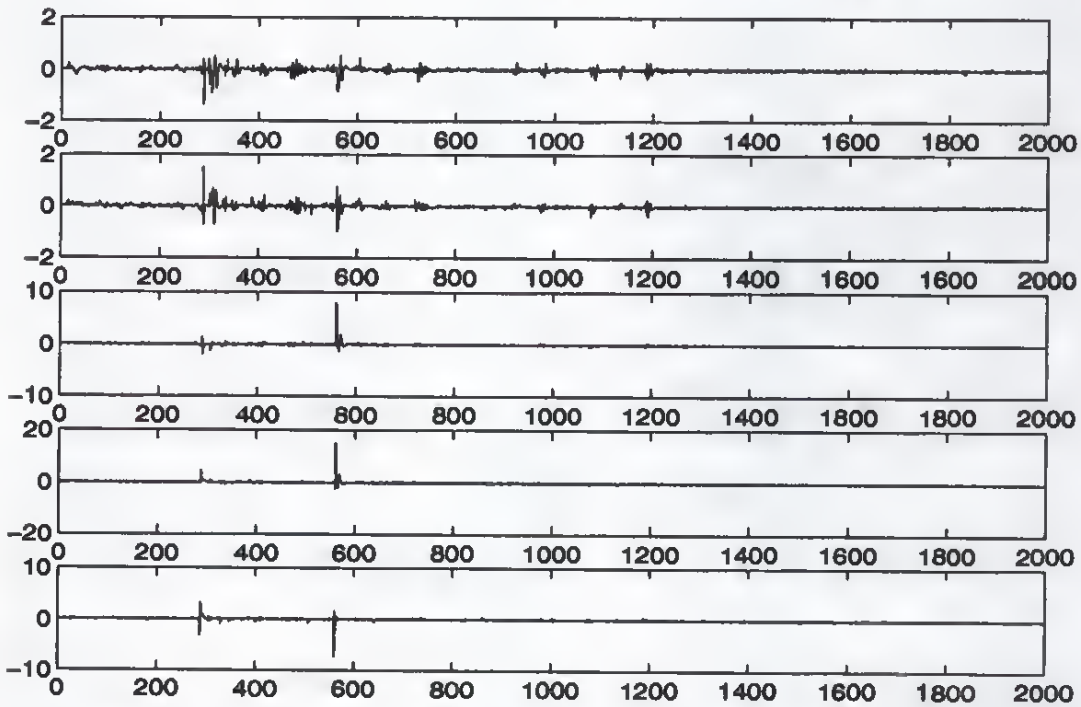


Figure 17. The last five PCA decompositions of the EEG signal with 10 delay units

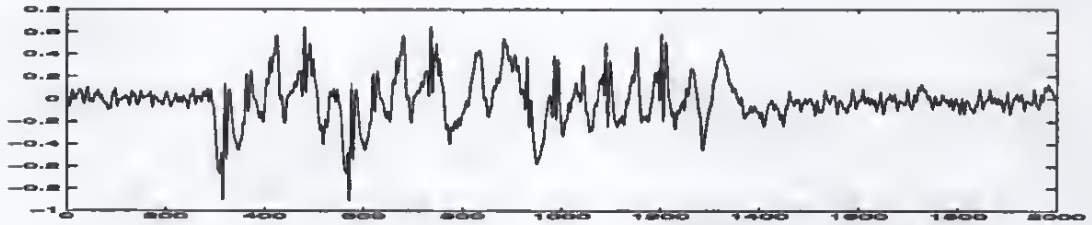


Figure 18. The reconstructed signal from the outputs of the PCA network with 10 delay units

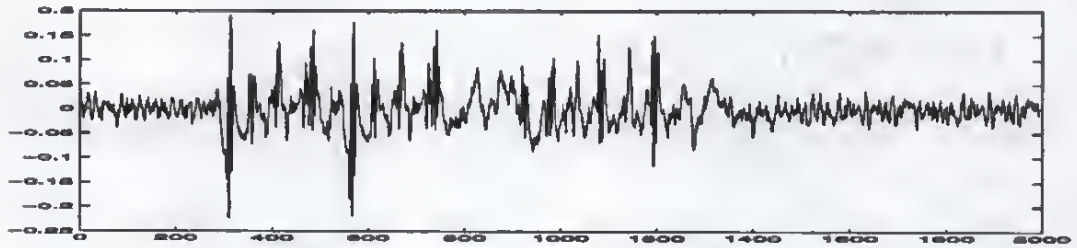


Figure 19. The error signal between the original and reconstructed signals with 10 delay units

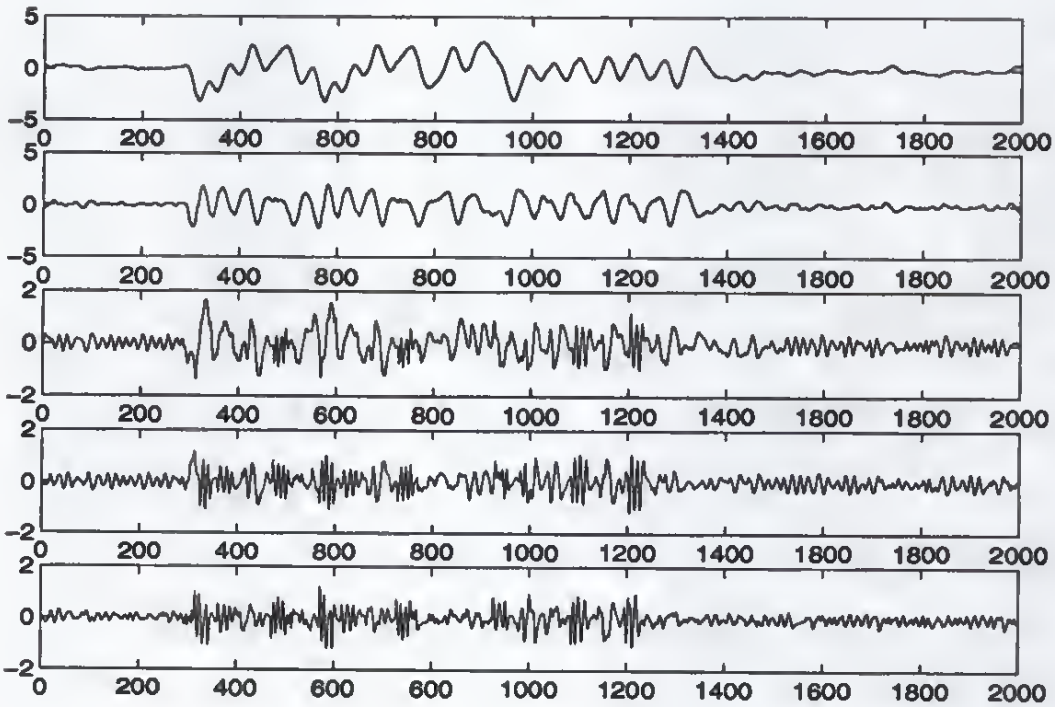


Figure 20. The first five PCA decompositions of the EEG signal with 30 delay units

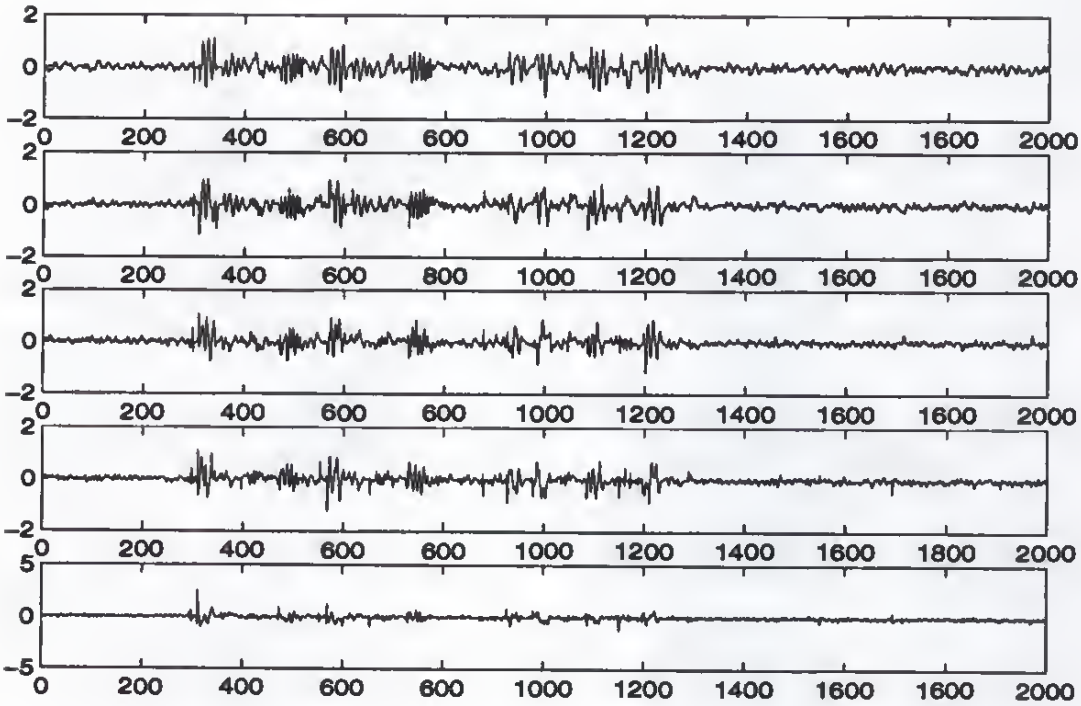


Figure 21. The last five PCA decompositions of the EEG signal with 30 delay units

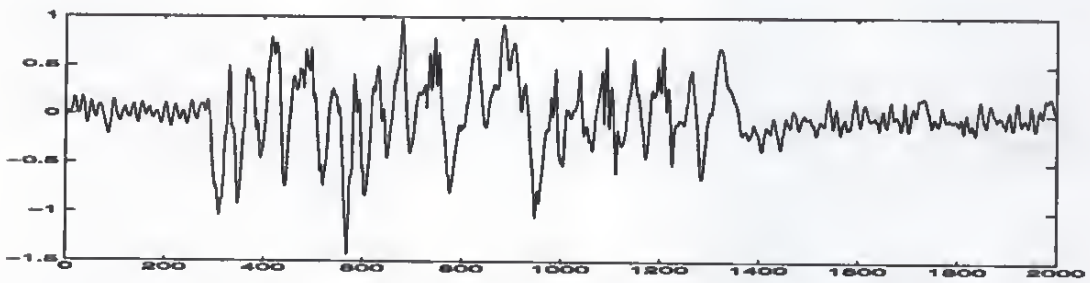


Figure 22. The reconstructed signal from the outputs of the PCA network with 30 delay units

Using 6 delay units in the input, the results are presented in Figures 23 and 24. The step size is 0.015. The MSE 0.0027 which is smaller than the MSE using 30 delay units but larger than the MSE with 10 delay units. The error signal is also similar to the one given in Figure 19.

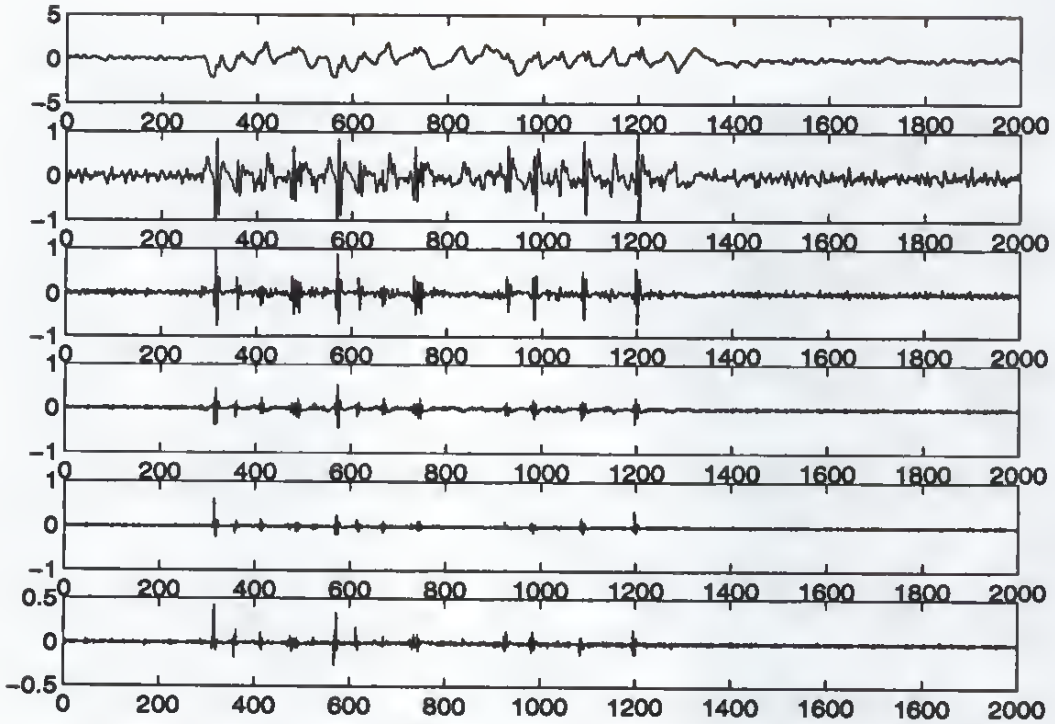


Figure 23. The PCA decompositions of the EEG signal with 6 delay units

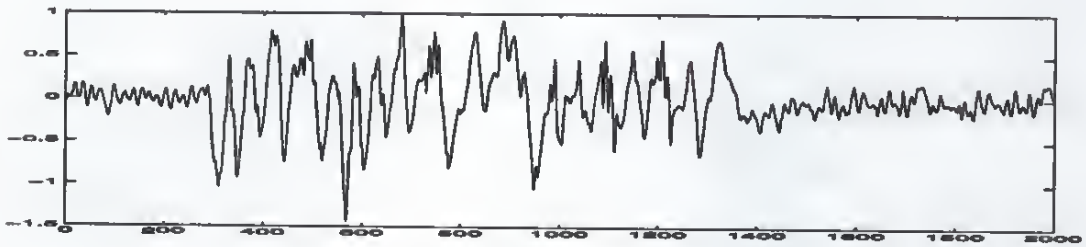


Figure 24. The reconstructed signal from the
outputs of the PCA network with 30 delay units

We also tried the larger window sizes such as 50 and 100, but the MSEs become larger with the increasing of window size. With 200 delay units, the four decomposed principal components and reconstructed signal is totally distorted as shown in Figure 25.

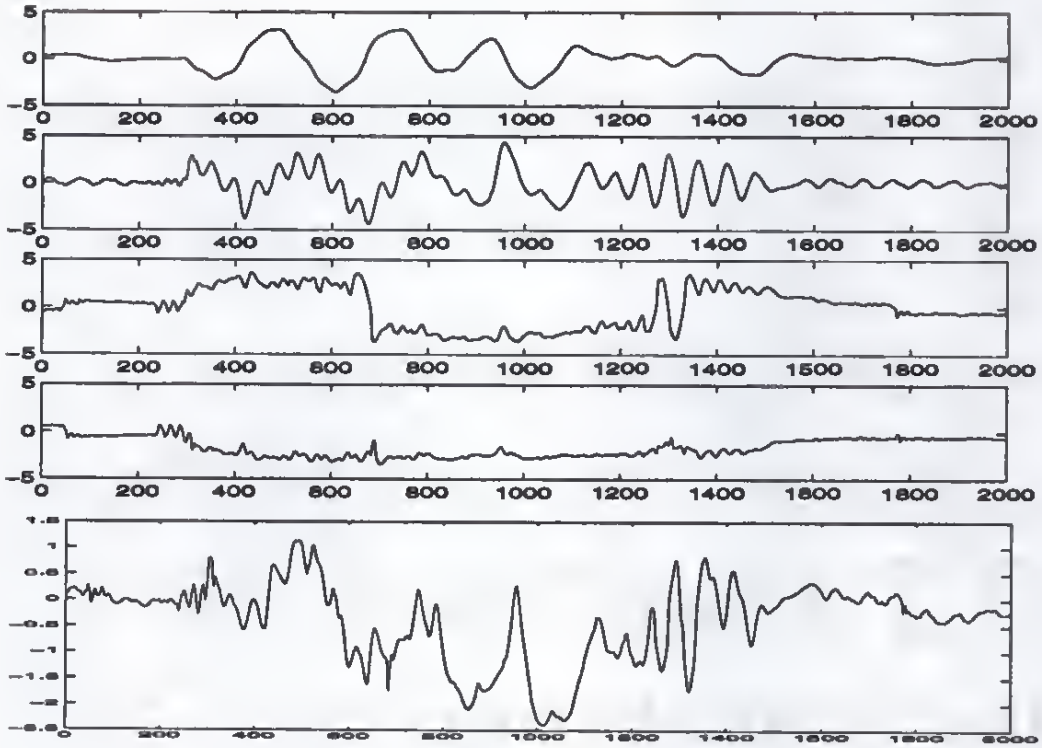


Figure 25. The first four principal components and reconstructed signal from the outputs of the PCA network with 200 delay units

From the simulation results, we can see that the temporal PCA network can process non-stationary signals such as EEG by using its time-frequency decomposition property. But the network should be very carefully designed regarding the window size.

It is very interesting to observe from the above examples (Figures 17, 21 and 23) that with the smaller window size the higher frequency spikes appear more dominantly than when using larger window size. This can be understood since the frequency resolution of the decomposition is closely related to the size of window. Furthermore, when the larger window is used, the decomposed signals look like sinusoids (Figures 20 and 21), which verify the argument given by theorem 3.3.1 that the temporal PCA will converge asymptotically to the Fourier transformation for finite stationary signal. With small window, the decomposition is far from sinusoidal but preserve even better the input signal properties.

This is the advantage of PCA over the Fourier analysis: No matter what is the size of the window, the basis functions are always being tuned to the available data. It should be pointed out that when the window becomes too larger, as shown in Figure 25, the locally stationary property of the EEG cannot be preserved. So the network averages the input and principal components do not show the sinusoid-like waveforms, and the reconstructed signal is distorted. Therefore, it is very important to select a proper window size for nonstationary signal processing.

3.6.2 Experimental Results for Multi-Resolution Analysis Based on the Temporal PCA

The multi-resolution network given in Figure 14 is used in this experiment. Ten delay units are included in the tapped delay line. And four output units are used for the reconstruction. The connections between the input and output units are 11, 9, 7, and 5 for the first, the second, the third, and the fourth output units, respectively. The step size is 0.01. After 25 iterations, the decomposed signals are shown in Figure 26. And the reconstructed signal based on the four signal given in Figure 26 is shown in Figure 27. The MSE between the original signal and the reconstructed signal is 1.52×10^{-6}, which is better than the result presented in section 3.6.1. The error signal between the original and the reconstruction signal is shown in Figure 28.

We also compare our multi-resolution network with the Daubechies wavelet [see details in Daubechies, 1992]. Four levels are used in Daubechies wavelet, and simulation results on various coefficients are given below. First, we used the same number of coefficients (33) in the wavelet as the multi-resolution network. The reconstructed signal is shown in Figure 29, and the MSE is 0.0196. With 50 coefficients, the MSE is 0.171, and the reconstructed signal is depicted in Figure 30. Comparing with the result (MSE=0.0152) from the multi-resolution network, we see that the wavelet even with more parameters gives worse results. Of course, with more coefficients in the wavelets, the MSE

could be decreased. As an example, 100 coefficients is used. The reconstructed signal is shown in Figure 31, and the MSE=0.00843. Another very critical point is that the reconstructed signals from the wavelet drop those samples with smaller amplitudes which may be very important for the dynamics of the whole signal, as can be seen from the error signal depicted in Figure 32. Therefore, we can said that the wavelet may blur the original signal. But the multi-resolution network keeps those smaller amplitude samples, so the dynamics of the reconstructed signal are better overall. Although using different way to select the coefficients may keep some coefficients corresponding to small amplitude samples, the MSE becomes much larger due to the distortion in large amplitude samples. Therefore, it is very hard to reconstruct the EEG signal with small coefficients even with different assignment of coefficients. It should be pointed out that both Daubechies wavelet and the temporal PCA can yield a perfect reconstruction with a large number of bases.

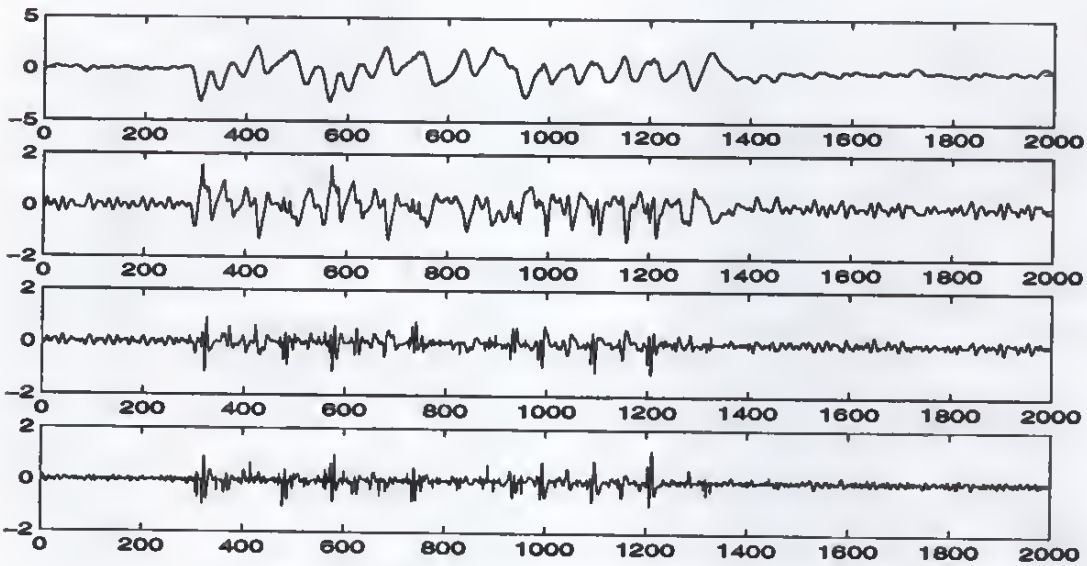


Figure 26. The first four multi-resolution decompositions of the EEG signal

From this section, we see some advantages of the multi-resolution network. Among them, the most important point is that the basis functions can be adjusted based on the statistical properties of the input signal yielding more accuracy than the Daubechies wavelet

for a small number of coefficients. So, this is a very powerful tool, especially for practical applications where most properties of the signal are usually unknown, and the environment is time-varying. One critical issue for the multi-resolution network is that further mathematical analysis is necessary, since we may not say, at this moment, that the network outputs are the principal components of the input signal. But this network works better than the others based on the experimental results.

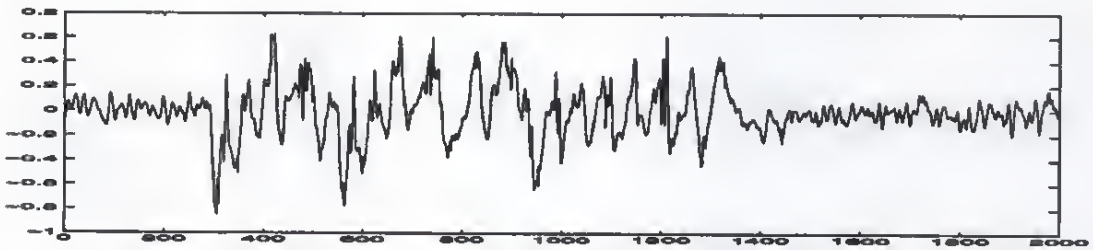


Figure 27. The reconstructed signal from the outputs of the multi-resolution network

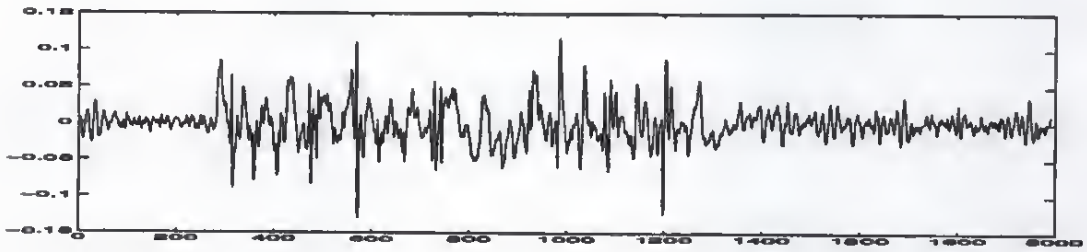


Figure 28. The error signal with multi-resolution network

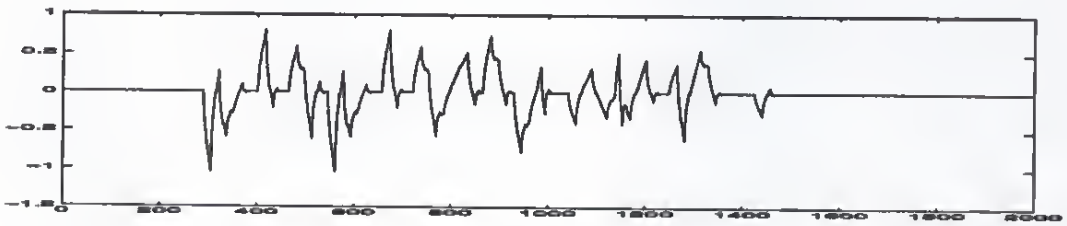


Figure 29. The reconstructed signal by the
Daubechies wavelet with 4 levels and 33 coefficients

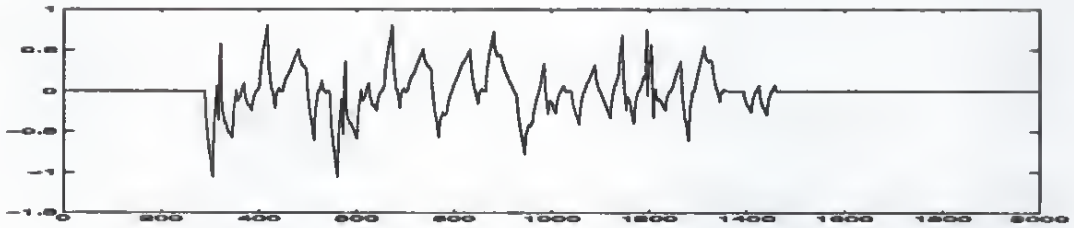


Figure 30. The reconstructed signal by the
Daubechies wavelet with 4 levels and 50 coefficients

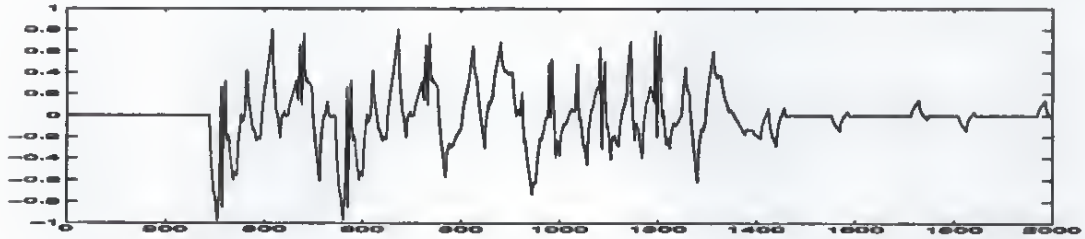


Figure 31. The reconstructed signal by the
Daubechies wavelet with 4 levels and 100 coefficients

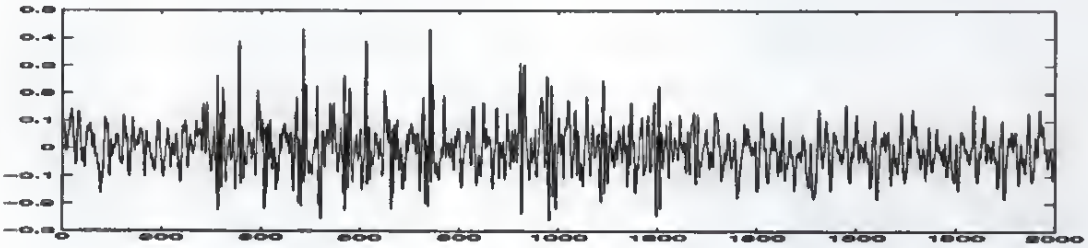


Figure 32. The error signal using
Daubechies wavelet with 4 levels and 100 coefficients

3.6.3. Transform Domain LMS Filter

3.6.3.1. Background

Adaptive filters have been widely studied by many researchers, and many structures and weight adjustment algorithms have been proposed [Haykin, 1996]. Among the family

of adaptive filters, the linear transversal filter with the LMS algorithm plays a fundamental role since it is very simple and very efficient to implement. One of drawbacks of the LMS algorithm is that the convergence becomes very slow when the eigenvalue spread or eigenvalue ratio, which is defined as the ratio of largest eigenvalue to minimum eigenvalue, is large [Widrow and Stearns, 1986; Haykin, 1996]. Another drawback is the redundancy between the signals at the taps of the LMS filter; namely, the signals between taps are highly correlated to each other. For the first problem, a technique called self-organizing was proposed, of which a whitening transformation is used to equalize all eigenvalues of the autocorrelation matrix of the input signal. The problems associated with the self-organizing algorithms are that they are not on-line algorithms and need a large number of arithmetic operations due to the matrix manipulations. In order to remove the second drawback of the LMS, fast Fourier transform (FFT), discrete Fourier transform (DFT), and discrete cosine transform (DCT) have been applied to orthogonalize the signals between taps. With these orthogonal transformations as preprocessors, the LMS algorithm is referred to transform domain LMS algorithm. For the clan of transform domain LMS algorithms, the problems are: (i) they are all N-by-N transformations, that is, no data reduction can be considered since the transformed signals are ordered based on frequencies, (ii) most of them are used in the block processing mode, not on-line mode, in order to save computation complexity, (iii) they are used only for time-invariant signals since FFT, DFT, and DCT are only valid for stationary signals. It is well know that Karhunen-Loeve Transform (KLT) would be the ideal transform because it takes into account the eigenvalues of the input signal correlation matrix [Shynk, 1992]. However, since this matrix is assumed to be unknown, the KLT can not be used in on-line algorithm.

It is well know that the KLT and PCA are very similar techniques [Gerbrands, 1981], and PCA can be implemented by on-line Oja or Sanger's rules [Oja, 1992; Sanger, 1989], therefore, the PCA learning can be used in transform domain LMS algorithm. Furthermore, data reduction can be done with temporal PCA transform since the outputs of the

PCA network are ordered based on the energy of the input signal or the eigenvalues of the signal correlation matrix. In order to track time-varying signal and also speed up the convergence rate of the original PCA algorithm, recursive estimation based on the Kalman filtering algorithm combined with search-then-convergence strategy is used to update the weight in temporal PCA transform.

3.6.3.2. Transform domain LMS algorithms

Transform domain LMS algorithms, which was proposed by Widrow et al., and Narayan et al. [Windrow et al., 1984; Narayan and Peterson, 1983], can be illustrated by Figure 33.

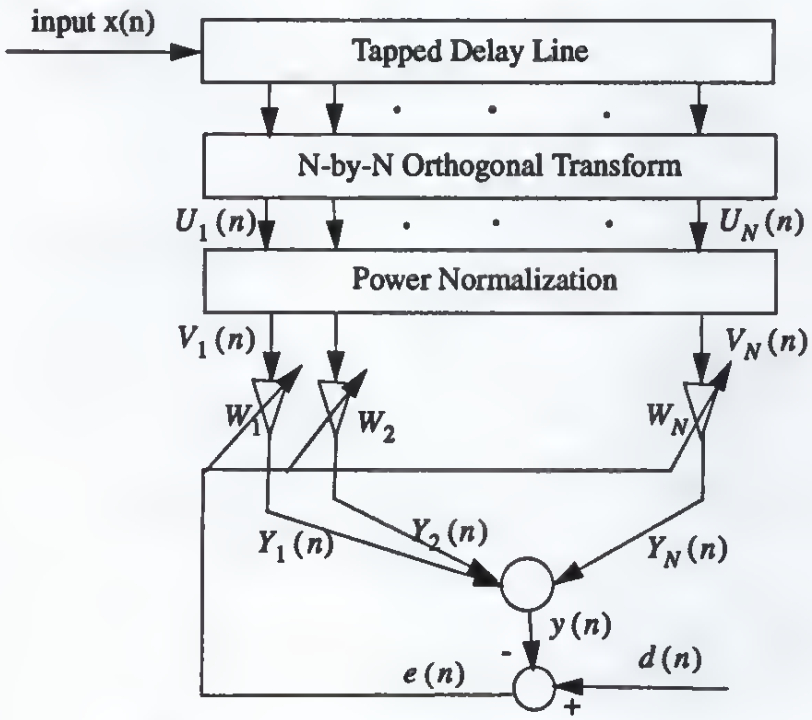


Figure 33. Conventional transform domain LMS filter

where d(n) is the desired signal, e(n) is the error signal, $U_i(n)$, $V_i(n)$, W_i and $Y_i(n)$, i=1,...,N, are transferred signals by theN-by-N orthogonal transform, the normalized

transferred signals by the signal power, the weights, and output elements in each channel in the transform domain, respectively.

The N-by-N orthogonal transform can be any orthogonal transformation, but FFT, DFT, and DCT are preferred because they can be applied in most real-time applications and fast algorithms are available.

The weights are adjusted by the classical LMS algorithm [Shynk, 1992]:

$$\Delta W_i(n) = 2\mu e_i(n) V_i(n) \tag{88}$$

where μ is the step size.

It should be pointed out that we do not care in what domain, (i.e., the frequency domain), the $V_i(n)$ are, since the error signal $e_i(n)$ can be always obtained in the time domain if the orthogonal transform is complete (the error between the original signal and transferred signal asymptotically goes to zero) [Rabiner and Gold, 1976; Davidson and Falconer, 1991].

3.6.3.3 Ideal transform domain LMS algorithm based temporal PCA learning

A new transform domain LMS filter is given in Figure 34, in which the only difference with the conventional transform domain LMS filter given in Figure 33 is that the PCA which is a N-by-M ($M \leq N$) transformation replaces the N-by-N transformation.

If M<N, data reduction can be done, which will improve the LMS training procedure.

The PCA network used here is slightly different from the conventional PCA network which does not include the tapped delay line and is applied to static pattern.

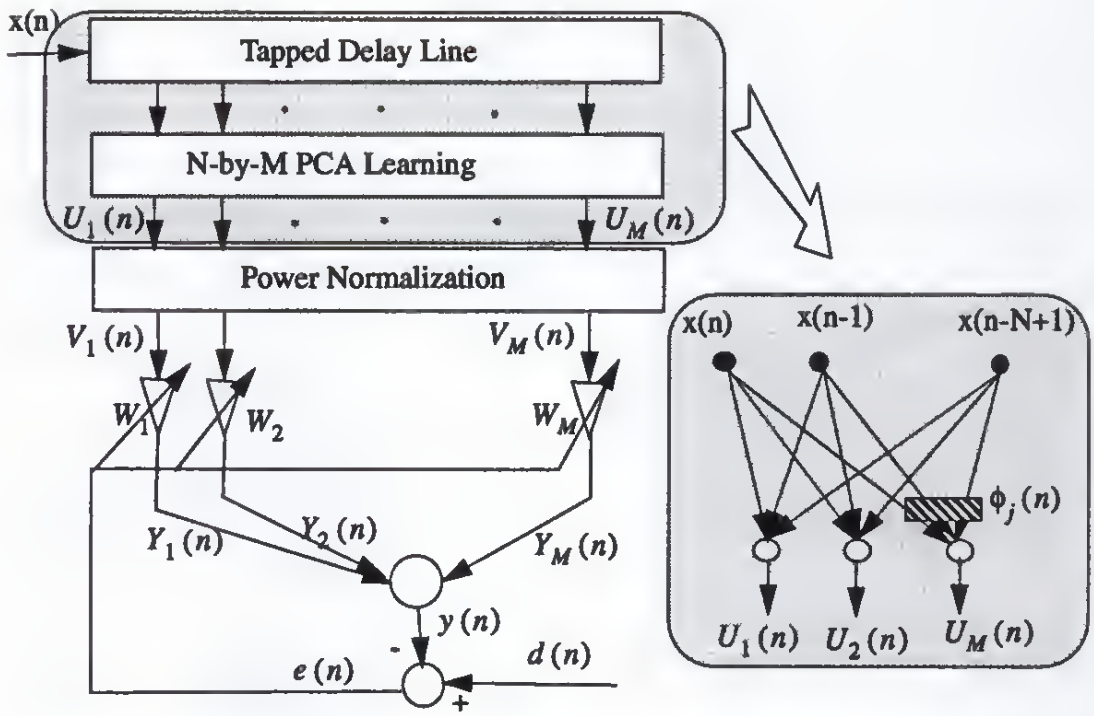


Figure 34. A transform domain filter based on the temporal PCA learning

The training algorithm adopted here is the recursive estimation based on Kalman filter, which has the form of [Bannour, 1995]:

$$\Delta\phi_j(n) = \mu_j(n) K_j(n) [X(n) - U_j(n)\phi_j(n-1)] \tag{89}$$

where X(n) denotes the signal vector after the tapped delay line, $\phi_j(n)$ is the weight vector of the PCA network connection to the output $U_j(n)$,

$$K_j(n) = \frac{P_j(n-1)U_j(n)}{[1 + U_j^2(n)P_j(n-1)]} \tag{90}$$

and

$$P_j(n) = [1 - K_j(n)\,U_j(n)]\,P_j(n-1) \tag{91}$$

$$P_j(n) = [1 - K_j(n)\,U_j(n)]\,P_j(n-1) \tag{92}$$

Notice that in Eq. (89), a step size μ_j is put before the gain $K_j(n)$,j=1,..,M, this makes the adaptation rule slightly different from the original Kalman filter approach proposed by [Bannour, 1995]. The step size will take account the search-then-convergence idea into the learning [Darken and Moody, 1992]; namely,

$$\mu_j(n) = \frac{\mu_{j0}}{1 + n/\tau_j} \tag{93}$$

where μ_{j0} are constants, and τ_j are also constants.

Due to the fact that the eigenvectors of the PCA converge sequentially from the largest one to the smallest one [Bannour, 1995; Sanger, 1989], properly selecting the parameter μ_{i0} and τ_j can reduce the computation requirement of the algorithm. That, we only adapt the weights connected to the first neuron in the output layer. After the weights converge, stopping the training for the weights and then adjusting the weights connected to the second neuron in the output layer. Following the procedure until the weights connected to the final neuron converge. Training the PCA in this way, the computation required is simply O(2N+M) instead of O(M*(2N+1) for simultaneously training. The computation complexity for the LMS is O(2M), so the total computation complexity is O((2N+3M)). And it is $O(K\log_2 K + 2N)$ for the FFT case. Obviously, the computation requirement for the PCA type LMS is less than the FFT type LMS. In our case N=10 and M=4, the PCA type LMS needs 32 algebraic operations, and the FFT type LMS needs 84 algebraic operations since K has to be 16.

3.6.3.4 Experimental results

One-step prediction problem for both stationary and nonstationary signals are considered here in order to show the advantages of the proposed approach.

First, let us deal with the stationary case. A stationary signal y(n), n=1,...,3000, is generated by using a white noise sequence p(n) passing through a first order AR model:

$$y(n) = 0.7y(n-1) + p(n) \tag{94}$$

where the initial value $y(0) = 0$.

Following parameters are selected for the simulation: N=10 and M=4.

The first four eigenvectors obtained from the PCA network are compared with the corresponding eigenvectors using eigen-decomposition directly on the autocorrelation matrix in Figure 35. The mean square errors (MSEs) are 0.00709, 0.00728, 0.02321, and 0.02384 for the first to the fourth, respectively. It can be seen that the PCA network gives very good estimations of the true eigenvectors.

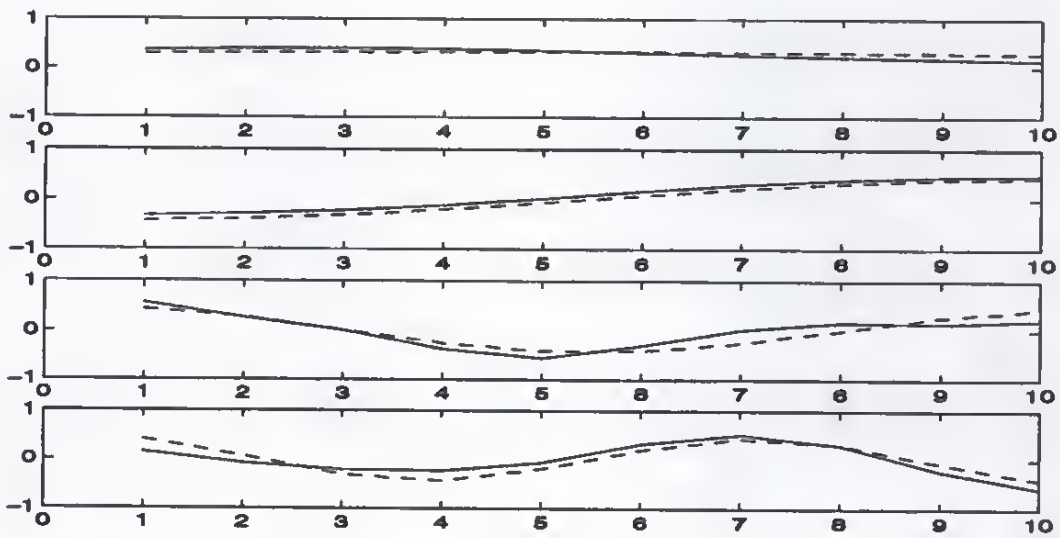


Figure 35. The eigenvectors of PCA (dashed-line) and their true values (solid-line)

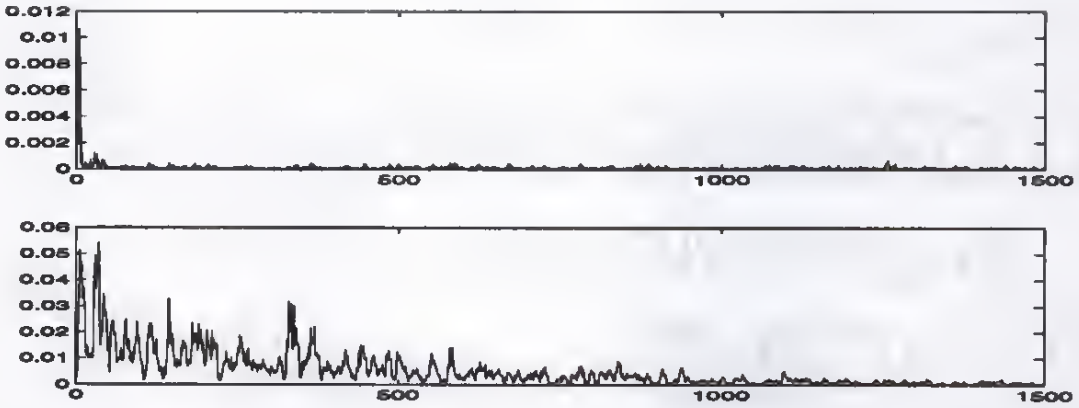


Figure 36. The learning curves of the transform domain LMS (upper) and LMS (lower)

Figure 36 presents the learning curves of the transform domain LMS and the classical LMS algorithms. For the classical LMS, 10 delay units are used to minimize mean square error, and for transform domain LMS, only 4 outputs of the PCA network are involved in the adaptation. Obviously, the transform domain LMS converges much faster than the classical LMS algorithm. The final errors are comparable, although only four channels are used in the transform domain LMS.

Next, a local stationary signal is used for simulations. A EEG signal segment is shown in Figure 37. And the corresponding learning curves for both transform domain and classical LMS are depicted in Figure 38. Clearly, the transform domain LMS results in very fast convergence rate.

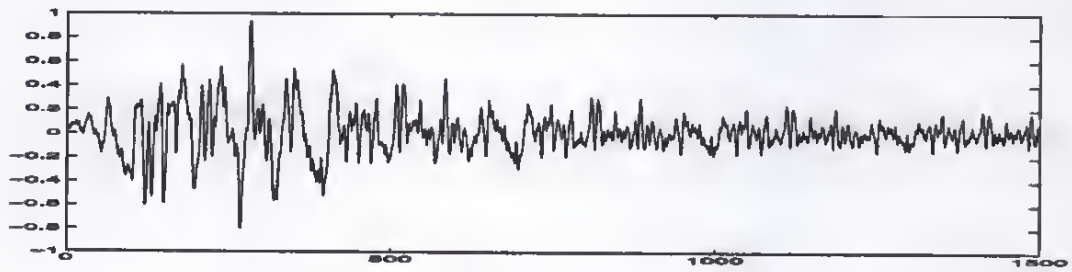


Figure 37. A EEG signal segment

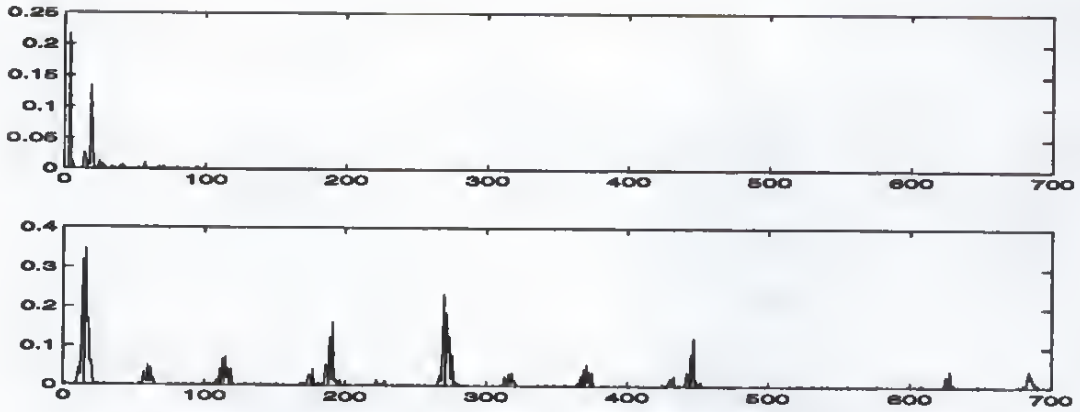


Figure 38. The learning curves of the transform domain LMS (upper) and LMS (lower)

Therefore, the transform domain based on temporal PCA provides not only fast convergence rate, but less computation complexity also.

The above discussions has been shown that this neural based approach provides an fast on-line training method for transform domain LMS algorithm. Moreover, less computation requirement are needed due to the fact that the outputs of the PCA are ordered based the energy (variances) of the output signals such that those signals with less energy can be ignored without affecting seriously the final MSE. In addition, some properties of the temporal PCA network such as the orthogonality, on-line tracking, and ordered output signals have been shown clearly in this experiment.

3.7 Discussions

This chapter extended the static PCA network to a temporal PCA network with the help of tap delay line. Mathematical analysis was presented to show that the temporal PCA network with a carefully selected window can extract time-frequency information for stationary signals, at the same time the other properties of the static PCA such as ordered output, orthogonal basis, and on-line adaptation are still kept. Moreover, we proposed a

multi-resolution network which is more power than the short- time-frequency network and wavelet based on our simulation results. In addition, the temporal PCA network was applied to the transfer domain LMS filter in order to build a ideal on-line transfer domain LMS filter, which clearly show the important properties of temporal PCA. It is important to point out that the tapped delay line used in this chapter can be replaced by Gamma or other short-term memory structures.

It is not difficult to see that the learning rule in the temporal PCA has the same form as that of static PCA. Therefore, the correlation based learning principle also works for the PCA in time network. Furthermore, the conclusions obtained in Chapter 5 can be extended straightforwardly to the temporal unsupervised networks case without re-deriving the corresponding formularies.

CHAPTER 4

TEMPORAL DECORRELATION AND ITS APPLICATIONS IN ADAPTIVE BLIND SOURCES SEPARATION

4.1. Introduction

It is known, from Chapter 3, that normalized Hebbian learning can be used to extract the principal components (PCA) of a given signal. Anti-Hebbian learning, which is formed by including a minus sign in the Hebbian rule, can generate minor components (MCA) of a given signal [Oja, 1992]. It is also well known that the PCA and MCA are very useful in a signal processing framework since the PCA and the MCA spaces correspond to the so-called signal and noise spaces, respectively. Therefore, they can be employed as building blocks in many signal processing problems based on subspace methods. For instance, PCA is used extensively in features extracting, data compression, and spectrum estimation, while the concept of MCA has been applied to statistical signal processing algorithms such as MUSIC, ESPRIT, and so on [Scharf, 1991]. Sample by sample (iterative) implementations of PCA and MCA may have an edge in some signal processing implementations where the analytic solutions are too expensive to implement.

Although both the BP and Hebbian learning are very useful in many practical cases, few studies address how to combine these two learning types to design new algorithms. In teacher forcing Hebbian learning [Williams and Zipser, 1989] the conventional post-synaptic activity to compute the Hebbian weight update is substituted by another signal not associated directly with the weighted input signal. This rule is an example combining the ideas from both supervised and unsupervised learning, since the new signal can be a

desired response. This learning rule is the basis of heteroassociation for static patterns, but as far as we know, it has not been explored in a signal processing framework. Motivated by this observation, we study in this chapter how to use teacher forcing Hebbian learning to compute the crosscorrelation between two signals. Diamantaras and Kung proposed a method to compute the crosscorrelation between two signals with the modified Hebbian rule, but it requires two static networks and different sets of weight [Diamantaras and Kung, 1994]. Instead of using two sets of weights and networks, we can simply use one dynamic network (i.e. a feedforward network extended with memory structures) and the teacher forcing Hebbian rule to obtain the same result. First, a very simple structure based on the transversal filter [Widrow and Stearns, 1985] will be proposed to compute the crosscorrelation function. Then we extend this idea to the IIR case using the Gamma filter [Principe et al., 1993] as an example. Finally, nonlinear extensions will also be considered since they can explore the independence between two signals.

As for the convergence of the proposed learning, an analysis is given through the new observation that the adaptation of FIR and IIR filters using LMS can be viewed as anti-Hebbian learning. Hence, the conclusion of convergence for LMS can be applied here also.

In terms of applications of the proposed new neural topology, we consider in this chapter the blind separation problem. Blind separation of multiple signal sources is very useful in many situations such as the "cocktail party effect" where microphones capture several simultaneous speakers, array signal processing, and noise cancellation with signal leakage. Recently, the blind separation problem has been studied by other researchers. Weinstein et al. proposed a frequency method for the two channels case [Weinstein et al., 1993]. They showed some very important results for decorrelation there, but their method cannot give analytic solutions without more constraints such as stationary and nonstationary. Van Grevan and Van Compernolle considered this problem from the angle of adaptive noise cancellation and used an intuitive cost function to do separation [Van Grevan and

Van Compernolle, 1995]. Using a neural method, Jutten and Herault proposed a nonlinear network to separate independent signals [Jutten and Herault, 1991]; Bell and Sejnowski applied the maximum entropy method to separate mixed signals [Bell and Sejnowski, 1995]; Matsuoka et al. minimized the correlation between the output signals [Matsuoka et al., 1995]. All of these works considered only signals instantaneously mixed, i.e. where the signal sources are multiplied by an unknown static mixing matrix. This is an unrealistic assumption due to the finite response time of any physical device. Furthermore, these methods are restricted to samplewise decorrelation which can be improved by using the signal information in a neighborhood.

We will propose a new separation method based on the temporal correlation idea mentioned above. The method is very elegant because it only requires uncorrelated input signal conditions for the linear temporal decorrelation networks, and input signal independence for the nonlinear temporal decorrelation networks. Moreover, it is highly suitable to separate locally nonstationary signals. Another important feature of our method is that decorrelation over time or temporal decorrelation is carried out instead of static decorrelation as proposed by Matsuoka et al. [Matsuoka et al., 1995]. It will also be shown that our method works very well not only for signals mixed instantaneously, but also when they are mixed over time. Therefore, it can be used for any signal separation problem.

The chapter is organized as follows: The analysis of teacher forcing Hebbian learning for FIR, IIR and nonlinear structures is given in section 4.2. A new structure for blind separation using temporal decorrelation will be presented in section 4.3. Section 4.4 presents the simulation results with speech signals and Gaussian noise. Section 4.5 contains the discussions.

4.2. Temporal Decorrelation by Teacher Forcing Anti-Hebbian Learning

4.2.1 Crosscorrelation can be Calculated with the Teacher Forcing Hebbian Rule

In this section, we will look at how the Hebbian rule can represent the temporal correlation between two signals. But let us define the crosscorrelation between two signals x(t) and d(t) first:

Definition (Crosscorrelation): The cross-correlation function between two signals at different time t_1 and t_2 is defined by [Papoulis, 1991]:

$$r_{xd}(t_1, t_2) = E[x(t_1)\, d(t_2)] \tag{95}$$

If both signals have sample length N+1, the full crosscorrelation matrix can be expressed by:

$$R_{xd} = E\begin{bmatrix} x(t)\,d(t) & x(t)\,d_1(t) & \ldots x(t)\,d_N(t) \\ x_1(t)\,d(t) & x_1(t)\,d_1(t) & \ldots x_1(t)\,d_N(t) \\ x_N(t)\,d(t) & x_N(t)\,d_1(t) & \ldots x_N(t)\,d_N(t) \end{bmatrix} \tag{96}$$

Estimating the crosscorrelation function is equivalent to estimating each element of R_{xd}. If the ergodic assumption is made, Eq. (6) can be utlized here. For practical purposes, L in Eq. (6) becomes the size of the time window.

In order to come up with a distributed, local, sample by sample implementation of the crosscorrelation function we propose the network shown in Figure 39 which we call a multi-FIR learning dynamics network. We refer to it as a learning dynamics network since its output is a set of quantities that can be used as weights for any neural network, in this particular case, according to the crosscorrelation function information between the neural

network input and output (Figure 40).

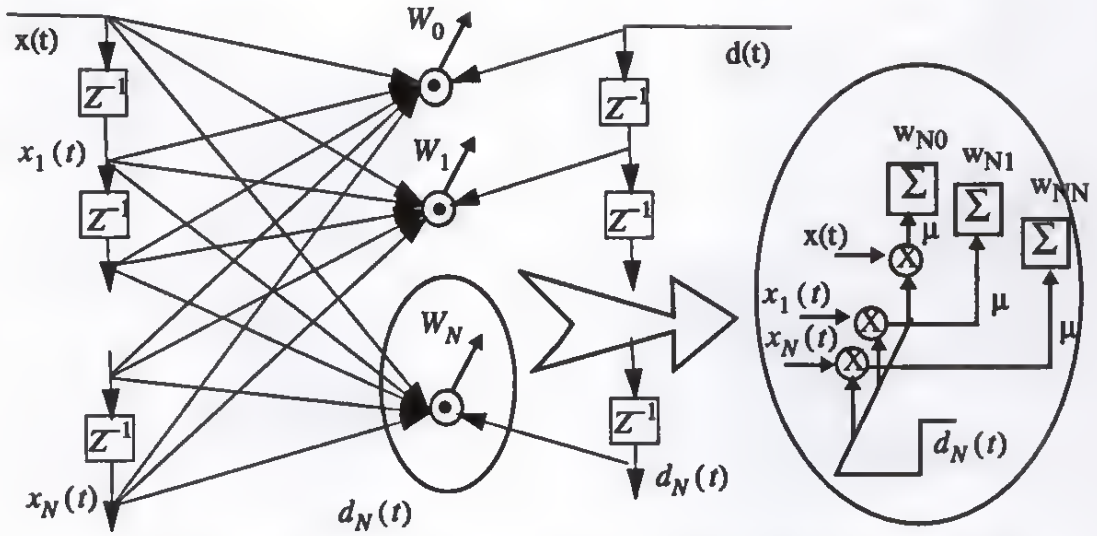


Figure 39. The learning network for computing the crosscorrelation matrix

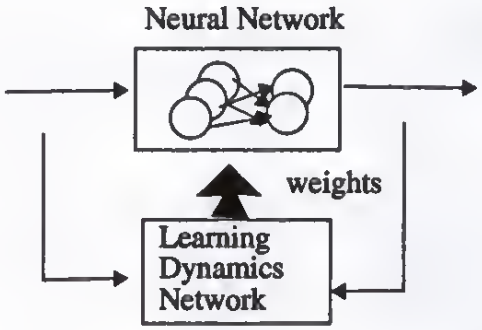


Figure 40. Neural network and the corresponding learning depicted as a learning dynamics network.

Normally we just write the learning equations without noticing that they can be also implemented as networks. This view is particularly important when deriving nonlinear learning equations, as will be exemplified latter. It can also save computation time since the value used for learning and to compute the input-output map are quite often the same.

In the learning dynamics network of Figure 39, the input x(t) and output d(t) are

passed through a delay line of N ideal delays each. For now, let us assume that the inputs are frozen in time. The delayed output samples $d_k(t)$ function as the postsynaptic signals for the input $x_j(t)$ so that the weight adjustment w_{kj} using the Hebbian principle can be expressed as

$$\Delta w_{kj} = \mu [d_k(t) x_j(t)] \tag{97}$$

where k,j = 0,1,...,N, and μ is the step size. The weight at iteration m will be

$$w^m{}_{kj} = w^{m-1}{}_{kj} + \mu [d_k(t) x_j(t)] \tag{98}$$

If the initial condition for the weight is set equal to zero, this can be written

$$w^m{}_{kj} = m\mu d_k(t) x_j(t) \tag{99}$$

For m=1/μ this value can be recognized as the k,j entry in the crosscorrelation matrix (Eq. 96).

In normal operation the input samples $x_j(t)$ and $d_k(t)$ are not constant, but are changing over time. So effectively

$$w^m{}_{kj} = \mu \sum_{t=1}^{m} d_k(t) x_j(t) \tag{100}$$

which can be recognized as proportional to the time estimates version of the crosscorrelation for lag k,j, i.e. for m=1/μ and dropping the limit operation

$$w^m{}_{kj} = A [d_k(t) x_j(t)] \tag{101}$$

So this shows that the network of Figure 39 produces as outputs estimates of the entries k,j of the crosscorrelation function between the input x(t) and the output signal d(t).

In order to make the notation more compact let us denote the signals drawn from the taps of the input delay line by the vector (vector quantities will be denoted by capital letters)

$$X(t) = [x(t), x1(t), ..., x_N(t)] \tag{102}$$

Let W_k, k=0,1,..,N represent the weight vector associated with $d_k(t)$

$$W_k = A[d_k(t)X(t)] = [w_{k0}\ w_{k1}\cdots\ w_{kN}] \tag{103}$$

that is, each output node of the network has dimension N+1 and estimates the kth column of the crosscorrelation matrix. The sample by sample adaptation rule is

$$\Delta W_k = \mu[d_k(t)X(t)] \tag{104}$$

It should be pointed out that the multi-FIR network given in Figure 39 is not redundant. For locally stationary signals (such as speech), the crosscorrelation matrix is non-symmetric. Even for stationary signals x(t) and d(t) since the cross-correlation matrix is a non-symmetric matrix, that is, $R_{xd}(-k) \neq R_{xd}(k)$, all the entree are necessary [Papoulis, 1991]. If the $x(t) = d(t)$, this operation corresponds to autoassociation. The autocorrelation function can be computed by the first column of Eq. (96), which corresponds to the first node of the network in Figure 39. But for locally stationary signals, the full network is again needed even to compute the autocorrelation.

Two final comments regarding this derivation are in order. Notice that Eq. (104) is implemented more appropriately in batch mode, but can also be implemented on-line.

This implementation of the Hebbian learning is straight forward to understand, but it is not very robust due to the unstable nature of the Hebbian rule. However, there are very well known procedures to stabilize the Hebb rule (see [Oja, 1992]) that can also be applied here, so we will not address this issue. Moreover, we are interested in utilizing the anti-Hebbian rule for decorrelation, which is stable as we will see later.

4.2.2 Temporal Decorrelation by Teacher Forcing Anti-Hebbian Rule

The general problem of decorrelation can be formulated as follows: Given a signal vector X(t) and a set of adjustable parameters W, construct a new signal vector D(t) from X(t), i.e.

$$D(t) = g(W, X(t)) \tag{105}$$

such that D(t) and X(t) are uncorrelated. Usually, g is a linear, continuous, differentiable function.

It is well known that the Hebbian rule represents the correlation between two signals and the anti-Hebbian denotes their degree of decorrelation [Foldiak, 1989; Oja, 1992]. We have already shown that a learning dynamics network using the teacher forcing Hebbian rule provides parameters that represent the crosscorrelation at different lags, hence, we can use teacher forcing anti-Hebbian learning for decorrelation. The rule for decorrelation reads

$$\Delta W_k = -\mu [d_k(t) X(t)] \tag{106}$$

where μ is the step size and k=0,1,...,N. Comparing with Eq. (106), this equation differs only in the minus sign, so the learning dynamics network to implement decorrelation has

also the topology of Figure 39.

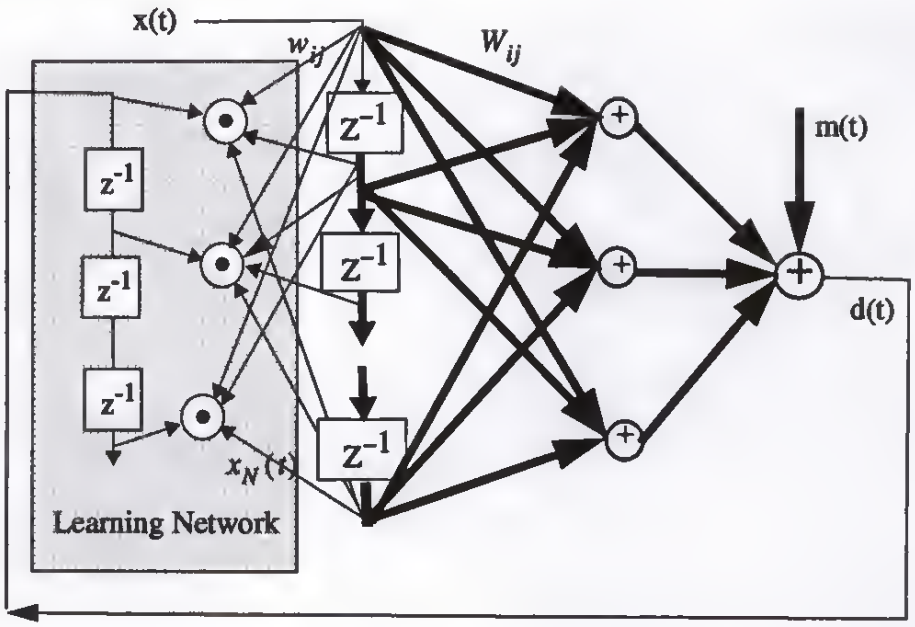


Figure 41. A temporal transversal filter decorrelator trained with a local anti-Hebbian learning network.

Up to now we only have addressed learning dynamics networks. An important aspect is how to use them in training linear or nonlinear neural networks. Figure 41 shows how the learning dynamics for decorrelation can be integrated with a transversal filter for a distributed, locally adaptive implementation of a temporal decorrelator. The bold lines represent the neural network and the shadowed box represents the learning network. The weights W_{ij} in the neural network are the copies of the corresponding weights w_{ij} in the learning network; namely, the weights w_{ij} of the learning network are adjusted based on the input x(t) and d(t), then they will be copied to the neural network for computing the new d(t). It is interested to see both the learning network and the neural network share the tapped delay line, such that the learning network and neural network are coupled together.

We have shown that the necessary condition for decorrelation is [Wang et al., 1994]

$$A[d_k(t)X(t)] = 0 \qquad k = 0, \ldots, N \tag{107}$$

Hence, decorrelation finds a signal D(t) in the null space of the input X(t).

4.2.3 Learning Dynamics Networks for Temporal Decorrelation Using IIR Filters

In the previous section the temporal decorrelation has been accomplished by means of the FIR structure. We can obtain similar results using the generalized feedforward filter class [Principe et al, 1993], which are normally IIR filters with local feedback. Here we demonstrate the concept with the Gamma filter due to its good stability property [Principe et al., 1993]. Similar to the structure given in Figure 39, the decorrelator which is referred as multi-Gamma using the Gamma filter is depicted in Figure 42.

The delay unit in the Gamma structure is:

$$G(Z,\lambda) = \frac{\lambda}{Z-(1-\lambda)} \tag{108}$$

where λ is the adjustable Gamma parameter. It has been shown that the stability of the Gamma filter can be guaranteed with the condition $0 \le \lambda \le 2$ [Principe et al., 1993], an advantage over the classical IIR filter. By analogy to the FIR case, the weight

$$\Delta W_k = -\mu[d_k(t)X(t)] \tag{109}$$

where

$$X(t) = [x_0(t), x_1(t), \ldots, x_N(t)] \tag{110}$$

$$x_k(t) = (1-\lambda)x_k(t-1) + \lambda x_{k-1}(t-1) \tag{111}$$

k=0,1,2,...,N, and

$$x_0(t) = x(t) \tag{112}$$

Since the first order recursive estimator can be considered equivalent to a window whose width is controlled by the parameter λ [Bellanger, 1987], it is straightforward to show that the weight matrix denotes an alternative estimation of the crosscorrelation matrix between x(t) and d(t).

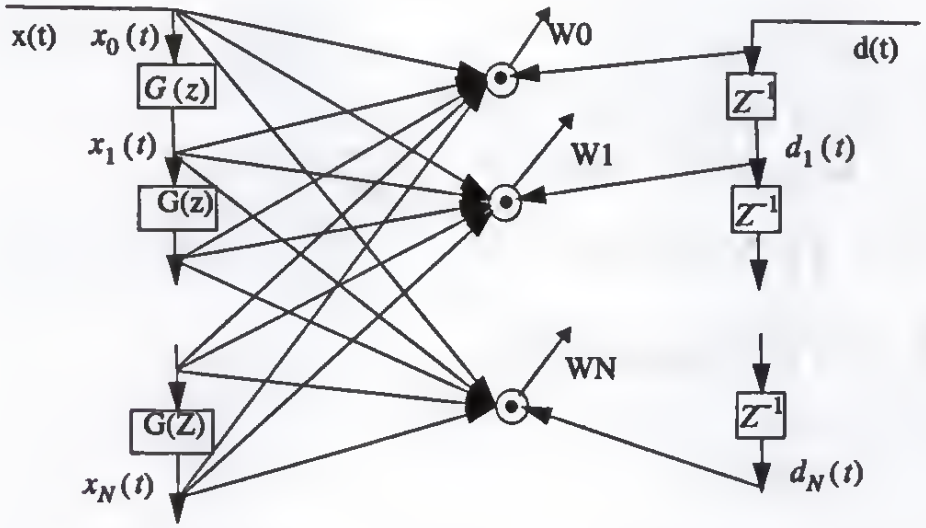


Figure 42. The learning dynamics decorrelator using the Gamma filter

Comparing with the FIR case, it is easy to show that the necessary condition for decorrelation is again

$$A[d_k(t)X(t)] = 0 \tag{113}$$

The adaptation rule for each weight vector leads to the same form as that of Eq. (100), except that

$$A[d_k(t)X(t)] = 0 \tag{114}$$

the delayed signals are replaced by the tap signals $x_k(t)$, k=0,1,...,N.

The Gamma parameter λ can be adapted with the following formula [Principe et al., 1993]

$$\Delta\lambda = -\mu_1 E\left[\sum_{j=0}^{N}\sum_{k=0}^{N} d_j(t)\, w_{kN}\alpha_k(t)\right] \tag{115}$$

where j denotes the nodes, and $\alpha_k(t) = \frac{\partial}{\partial\lambda}x_k(t)$ is computed as [Principe et al., 1993]

$$\alpha_k(t) = (1-\lambda)\,\alpha_k(t-1) + \lambda\alpha_{k-1}(t-1) + [x_{k-1}(t-1) - x_k(t-1)] \tag{116}$$

One of the advantages of the Gamma filter is that the temporal decorrelation can be done using a window adapted to the signal statistics, by means of adapting the parameter λ with the output mean square error.

4.2.4 Learning Dynamics Networks for Temporal Independence by Nonlinear Filters

Linear systems such as FIR and IIR filters can only utilize correlation information, which is equivalent to the second order input signal statistics. In some applications such as independent component analysis, correlations of higher order are necessary. It is known that independent analysis can be realized if all higher-order statistics are utilized [Comon, 1994]. Therefore, it is critical to extend the correlator learning networks to include higher order statistics. The most direct way is to work on the probability space based on the definition of independence given by [Papoulis, 1991]:

Definition (Independent processes): If two processes x(t) and d(t) are such that the

random variables $x(t_1), \ldots, x(t_N)$ and $d(t'_1), \ldots, d(t'_N)$ are mutually independent, then, these processes are called independent. Within probability space, this is equivalent to

$$p(X(t)D(t)) = p(X(t))p(D(t)) \tag{117}$$

that is, the joint probability of the two processes is equal to the product of each individual one,

$$p(X(t)D(t)) = p(X(t))p(D(t)) \tag{118}$$

where X(t) and D(t) denote the samples spaces of x(t) and d(t), respectively. The problem of directly using Eq. (118) to test independence is that the joint probability and each individual probability have to be estimated. This is not an easy task, especially for adaptive on-line algorithms since calculating probabilities needs all samples of training data.

Another approach for independent analysis explores the information contained in a finite number of higher-order statistics instead of using the probability density functions. Actually, in most practical cases, orders up to four have been shown sufficient [Nikias and Mendel, 1993].

In adaptive signal processing with neural networks, the nonlinearity is an alternate powerful tool for independent analysis. For example, Jutten and Herault included nonlinearities in their system and showed that the network was performing independent analysis. They used a static network so the topological placement of the nonlinearities was not so important. Since we utilize dynamic networks throughout this chapter, the location where the nonlinearities are included affect greatly the performance of the whole system. In our approach the nonlinear dynamic learning system for independent analysis is illustrated in Figure 43, where the nonlinearity f(.) is assumed continuous, differentiable, and bounded. The logistic function or hyperbolic tangent (tanh) function are good candidates, and p(t)

and q(t) denote the signals at the output of the corresponding nonlinearities.

The signals in the delay taps are

$$P(t) = [p(t), p_1(t), ..., p_N(t)] \\ = [f(x(t)), f(x_1(t)), ..., f(x_N(t))] \tag{119}$$

and

$$Q(t) = [q(t), q_1(t), ..., q_N(t)] \\ = [f(d(t)), f(d_1(t)), ..., f(d_N(t))] \tag{120}$$

for signals x(t) and d(t), respectively.

The weight vectors are still computed by teacher forcing anti-Hebbian rule, that is,

$$\Delta W_k = -\mu A [q_k(t) P(t)] \tag{121}$$

where k=0,1,...,N.

Therefore, the crosscorrelation matrix represented by weight vectors can be written as:

$$R_{pq} = A \begin{bmatrix} f(x(t))f(d(t)) & \dots & f(x(t))f(d_N(t)) \\ f(x_1(t))f(d_1(t)) & \dots & f(x_1(t))f(d_N(t)) \\ f(x_1(t))f(d_N(t)) & \dots & f(x_N(t))f(d_N(t)) \end{bmatrix} \tag{122}$$

It is straightforward to show that each element of R_{pq} contains higher order moments. As an example, let us expand in Taylor series the element $A[f(x_j(t))f(d_k(t))]$,

$$A\left[f(x_j(t))\,h(d_k(t))\right] = A\left[\left(\sum_{i=0}^{M}\frac{1}{i!}x^i(t)\frac{\partial^i f}{\partial x}\right)\left(\sum_{i=0}^{M}\frac{1}{i!}d^i(t)\frac{\partial^i f}{\partial d}\right)\right] \qquad (123)$$

where M is the number of terms truncated in the Taylor expansions. Obviously, higher orders are included in Eq. (123), therefore, independence can be realized approximately. Basically, the above analysis is an extension of the H-J method [Jutten and Herault, 1991].

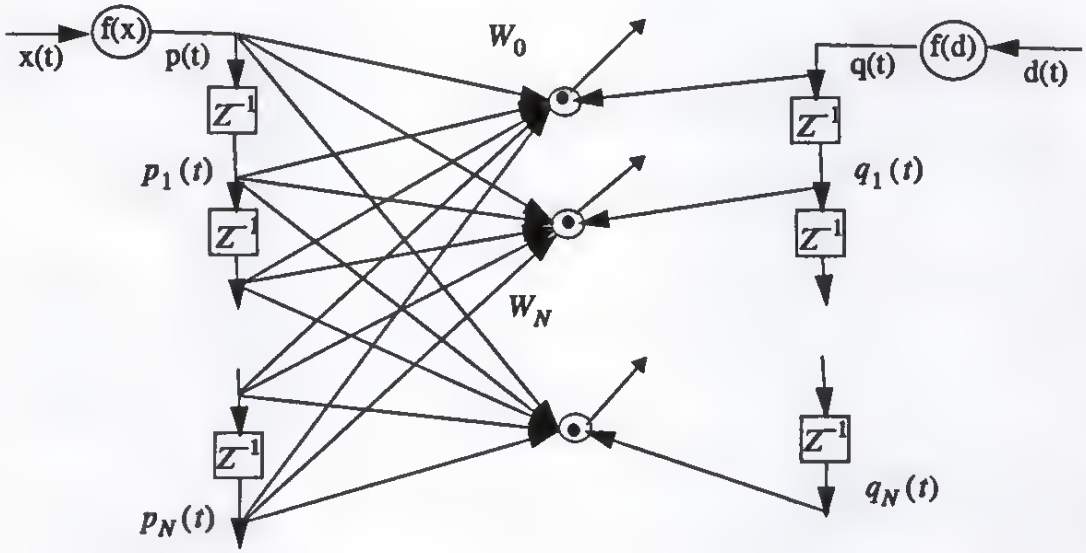


Figure 43. The nonlinear learning dynamics network for independent analysis

4.2.4 Stability of Teacher Forcing Anti-Hebbian Learning

In this subsection, we will first analyze the adaptive transversal or FIR filter trained with LMS criterion, and point out that the LMS learning can be viewed as a teacher forcing anti-Hebbian learning between the input signal and the negative of the output error. Based on this results, the stability of the teacher forcing anti-Hebbian learning is inferred from that of LMS.

Let us consider the basic transversal adaptive filter as depicted in Figure 44, where x(t) is the input signal, y(t) is the output signal, d(t) is the desired signal, e(t) is the error

signal which is defined as the difference between d(t) and y(t), and t is the discrete time index. The order of the FIR filter, or the number of tap delays, is denoted by N.

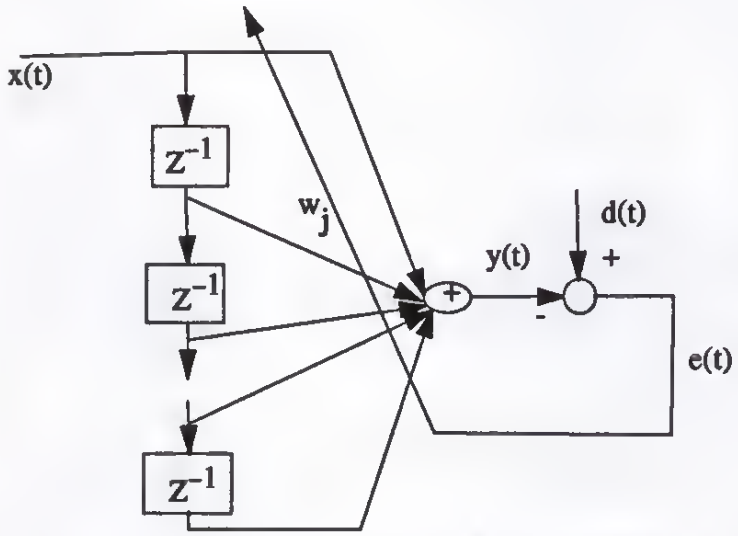


Figure 44. The transversal FIR filter

The cost function of the filter can be defined in the LMS sense as [Widrow and Stearns, 1985]:

$$J = e^2(t) = (d(t) - y(t))^2 \tag{124}$$

where

$$e(t) = d(t) - y(t) = d(t) - X(t)\,W(t) \tag{125}$$

and

$$y(t) = X(t)\,W(t) \tag{126}$$

The adaptation rule for the weight W is:

$$\Delta W = 2\mu e(t) X^T(t) \tag{127}$$

Eq. (127) corresponds to a Hebbian-like teacher forcing update for the weight W in terms of input/error relationship. Since the relationship between the error and the input can be formulated by $e(t) = -X(t) W(t) + d(t)$ which clearly shows the 'inhibitory' connection between the input and the error, Eq. (127) implements the anti-Hebbian rule. When the training is finished we have

$$E[e(t) X^T(t)] = 0 \tag{128}$$

Eq. (128) is exactly the Principle of Orthogonality in the adaptive FIR filter [Haykin, 1991]. Therefore, the anti-Hebbian teacher forcing learning which takes the form of Eq. (127) is rooted in adaptive supervised filter theory. Therefore, we can conclude that with this relationship given by Eq. (125) (which is the same as that in blind separation network, see section 4.3 for details), the stability of the teacher forcing anti-Hebbian rule can be guaranteed for neural networks if the step size of learning satisfies $0 < \mu < \frac{1}{\lambda_{max}}$ where λ_{max} is the largest eigenvalue of the autocorrelation matrix of the input signal.

4.3. Blind Sources Separation by Temporal Decorrelation

4.3.1 Blind Sources Separation Problem and Decorrelation Based Solution

Basically, blind separation problem can be stated as follows: A set of sources are mixed linearly by an unknown transfer function H, as depicted in Figure 45. The transfer function H can be a linear system or a constant matrix. Only statistical knowledge about

the sources S such as distributions, correlation, and independence is available. We collect the mixed signals X, and assume that their number is equal to the number of sources. Hence, the goal of blind separation is to design a system G, which can be either linear or nonlinear, to recover the original signals S from the observations X.

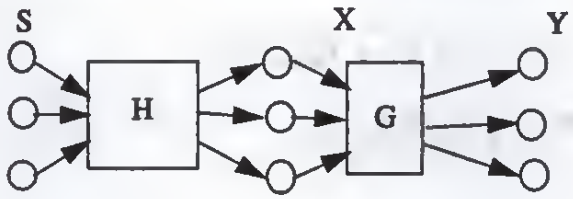


Figure 45. The blind separation problem

Mathematically, the above statement can be formulated as: In a linear system

$$X = HS \tag{129}$$

the transfer function H and the input signal S are unknown. The goal of blind separation is to find an optimal estimation

$$\hat{Y} = S \tag{130}$$

under a criterion with output signal X alone.

In this chapter, we deal with the signals mixed by a linear transversal filter (Figure 46), where H_1 and H_2, the mixing filters of the FIR type. Note that this mixing includes the instantaneous mixing as a special case (zero order filter). We don't assume that the delays of $s_i(t)$, i=1,2 are mixed with the sources which would lead to the much harder blind deconvolution problem. However, we believe that the inclusion of delays in the mixing matrix is much more realistic to model the cocktail party effect, and a necessary step to

fully understand the blind deconvolution problem.

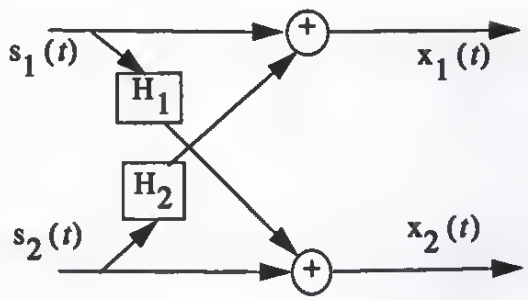


Figure 46. The original sources mixed by linear FIR filters

The problem we want to solve is recovering or estimating the original source signals $s_1(t)$ and $s_2(t)$ when only the linear mixed signals $x_1(t)$ and $x_2(t)$ are given.

The decorrelation system used is given in Figure 47, where G_1, and G_2, the decorrelation filters are the coupled structure given in Figure 41. The $x_i(t)$ and $d_i(t)$, i=1,2, correspond the m(t) and d(t) of Figure 41, respectively.

Decorrelation based blind separation problem has been studied by several research groups as mentioned in section 4.1 of this chapter. Weinstein et al. investigated various linear decorrelators and showed the possible solutions under certain conditions [Weinstein et al., 1993]. Van Gerven and Van Compernolle presented a detailed analysis of decorrelation using FIR filters. With a constant G_1, and G_2, which is called a scalar filter, Jutten and Herault and Matsuoka et al. showed good results for synthetic and speech signals [Jutten and Herault, 1991; Matsuoka et al., 1994]. We are not going to go into the details of those methods since the goal of this section is to show that blind separation based on the fully use of the crosscorrelation can improve the separation results.

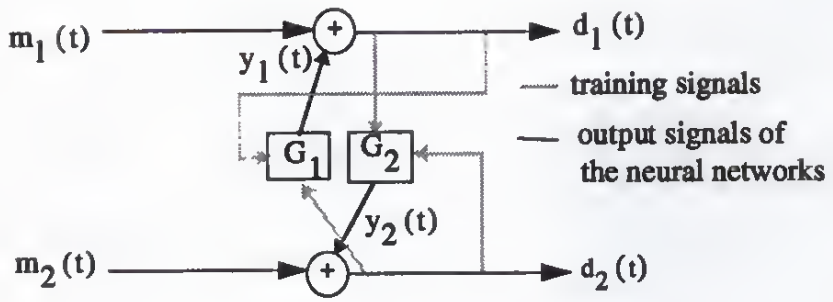


Figure 47. The decorrelation network

4.3.2 Experimental Results

In this section, the decorrelation network given in Figure 47 is used to decorrelate two uncorrelated signals. Speech signals show that the method works for nonstationary case, and stationary Gaussian noises show that this scheme works for the Gaussian case, too.

First, let us compare the decorrelation performance of decorrelating filters trained using three types of dynamical filters (multi-FIR, Gamma, and nonlinear) discussed in section 4.2 with FIR and scalar filters. We select two segments of two independent conversations (each with 20000 samples) from the TIMIT database for training and testing, which are shown in Figure 48. In the training stage, only 4000 samples (Figure 49) of them are used, since we think that them are enough to represent statistics of the speech signals. In the testing stage, the full signals (20000 samples) are utilized to address the issue of generalization.

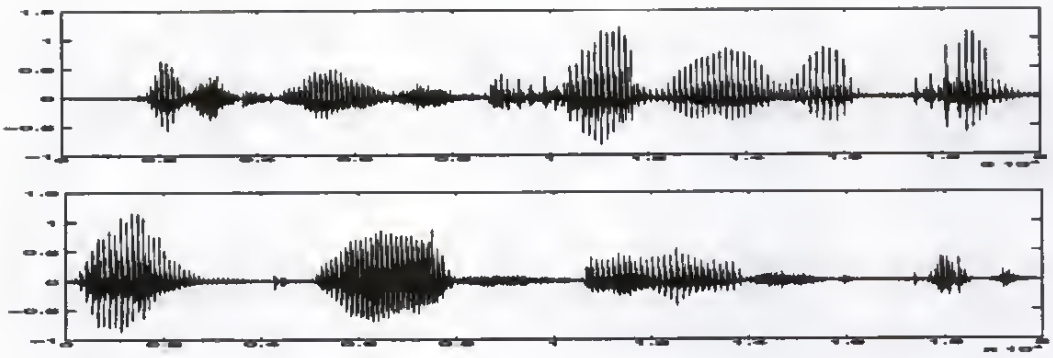


Figure 48. Two segments of speech signals

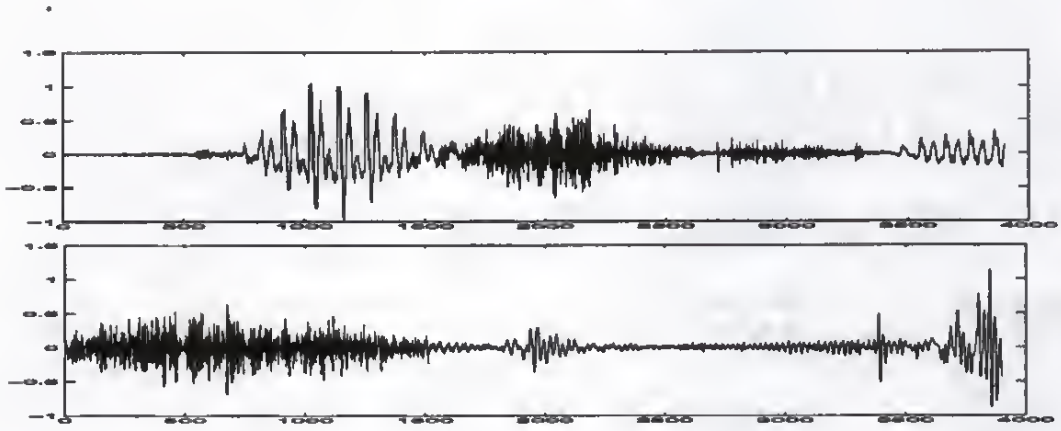


Figure 49. Two training speech signals

We will experiment with two kinds of mixing. One is called instantaneous mixing, that is, the mixed signals $x_1(t)$ and $x_2(t)$, generated from $s_1(t)$ and $s_2(t)$, are created by $x_1(t) = a_1 s_1(t) + a_2 s_2(t)$ and $x_2(t) = b_1 s_2(t) + b_2 s_1(t)$, where the mixing coefficients a_i and b_i, i=1,2, are selected based on the assumption $a_1 \geq a_2$ for $x_1(t)$ and n $a_2 \geq a_1$ for $x_2(t)$ since the distances between the speakers and microphones are usually different. Another one is mixing in time, namely, $x_1(t) = a_1 s_1(t) + a_2 s_2(t) + a_3 s_2(t-2)$ and $x_2(t) = b_1 s_2(t) + b_2 s_1(t) + b_3 s_1(t-3)$ where a_i and b_i, i=1,2,3, are selected based on the same assumption of mixing instantly. Instantaneous mixing is a special case of mix-

ing in time. One example of two signals mixing in time is shown in Figure 50. The mean of the mixed signals are both zero, and the variances are 0.0139 and 0.0109, respectively, for instant mixing; and 0.0224 and 0.0260, respectively, for mixing in time. The signals separated by the mixing in time algorithm are given in Figures 51 and 52, respectively.

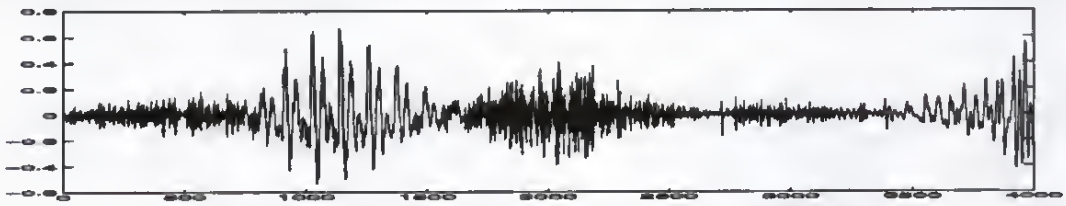


Figure 50. One example of mixed in time training signals

The criterion used to measure which learning methodology gives better results is the crosscorrelation coefficient between the two separated signals, since the goal is to decorrelate them. Mean square error is not a proper criterion since the shrinking in amplitude and delay in phase of the signals do affect the MSE, but not the quality of the separated speech. The change of the crosscorrelation coefficient during iteration, which is referred to learning curve, for both multi-FIR and FIR filters are given Figure 53. The final crosscorrelation coefficients are 0.0051 and 0.0060 for multi-FIR and FIR, respectively. It can be seen that the multi-FIR is outperforming the FIR filter with a smaller crosscorrelation coefficient.

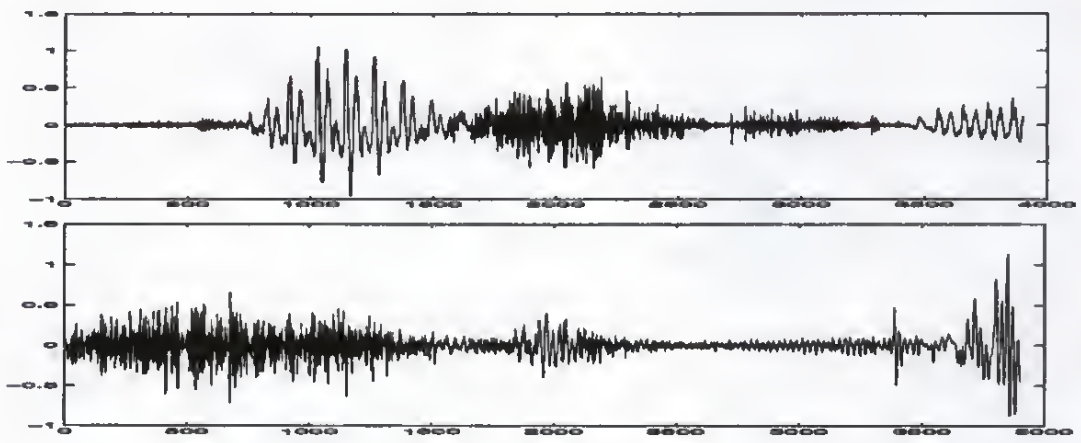


Figure 51. The separated signals using multi-FIR filters after training

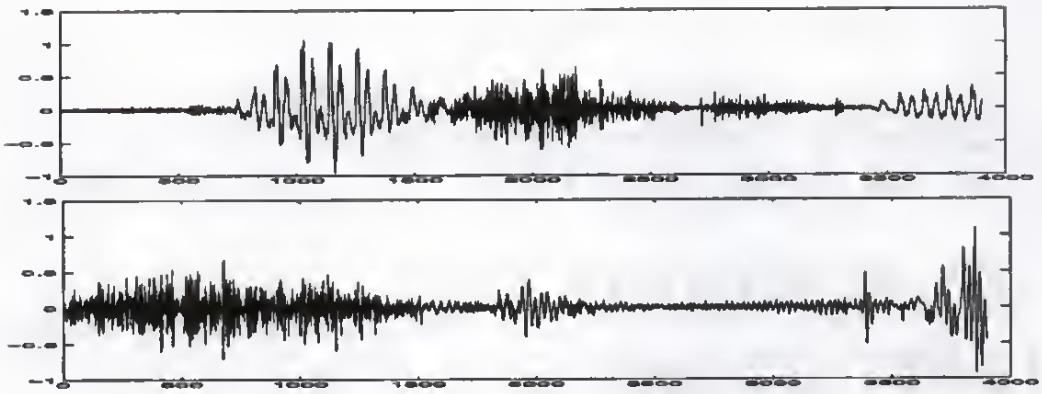


Figure 52 The separated signals using FIR filters after training

The above results only compared the abilities of FIR and multi-FIR decorrelators. With other decorrelators such as the scalar, Gamma, and nonlinear, training results are summarized in Table 1 which provides the final crosscorrelation coefficients in the training phase for mixing in time and mixing instantaneously. And the corresponding learning curves are given in Figure 53 and Figure 54, respectively. It should be pointed the order of decorrelators is selected based on the way of how the signals to be mixed. In the instantaneously mixing, the order can be lower; and for the mixing in time the order should be higher since the impulse response of a FIR filter which corresponds to mixing in time is longer that of the scalar which corresponds to mixing instantaneously. And the Gamma which is an IIR should have lower order than FIR or multi-FIR. Experimentally, similar results can be obtained with different orders.

Now, let us see what are the test results. In the testing stage, the trained weights are frozen. Then, the full signals with 20000 samples are mixed using the same mixing coefficients as the training signals. One example of the mixing in time signals is given in Figure 55. Feeding the mixed signals into the multi-FIR decorrelator, the reconstructed signals are shown in Figure 56. The results for multi-FIR and other decorrelators are presented in Table 2.

Table 1. Crosscorrelation coefficients for speech signals (training)

comparable parameters / names of decorrelator	Final crosscorrelation coefficient for mixing instantly	Final crosscorrelation coefficient for mixing in time
Multi-FIR	5.45×10^{-4} (Order 3)	5.10×10^{-3} (Order 6)
FIR	6.60×10^{-3} (Order 3)	6.00×10^{-3} (Order 6)
Gamma	6.60×10^{-3} (Order 2)	5.70×10^{-3} (Order 5)
Nonlinear FIR	6.80×10^{-3} (Order 3)	6.60×10^{-3} (Order 6)
Scalar	7.50×10^{-3}	9.79×10^{-2}

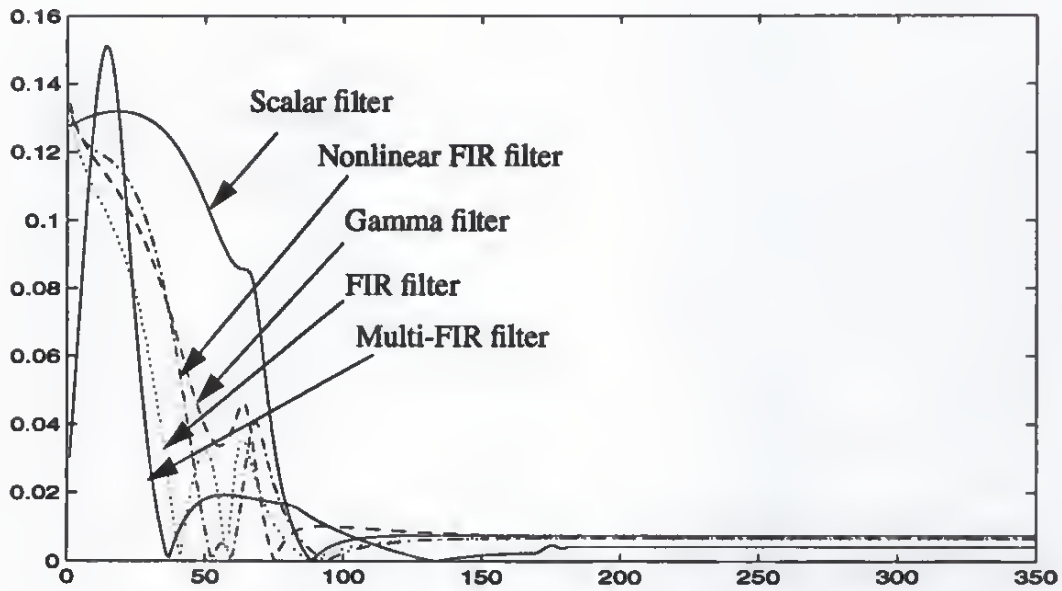


Figure 53. Learning curves for mixing in time signals

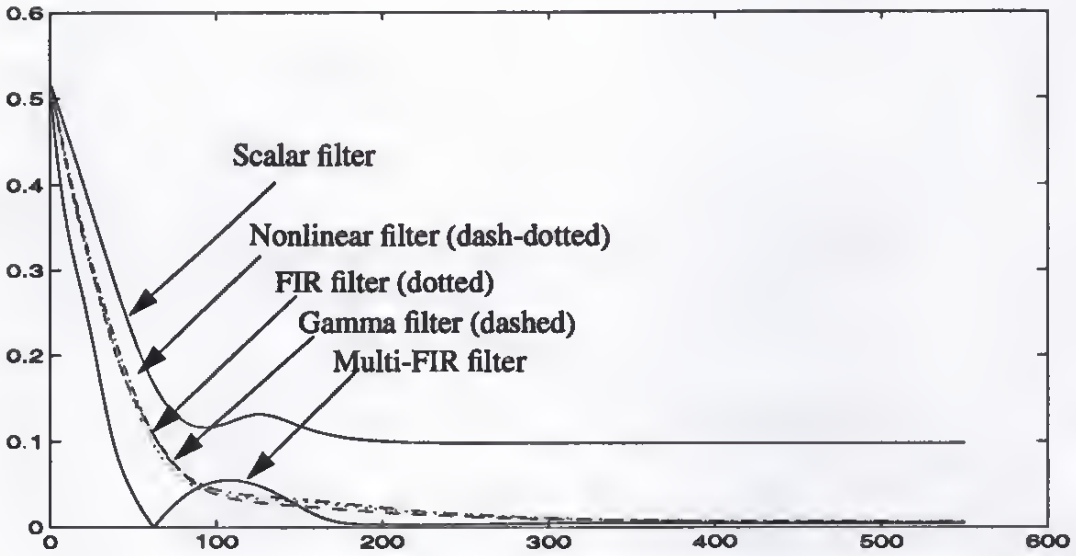


Figure 54. Learning curves for mixing instantly signals

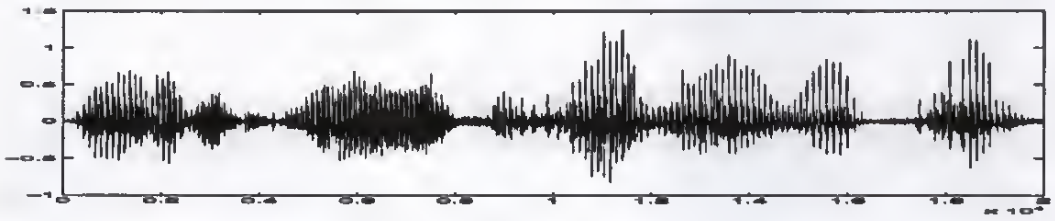


Figure 55. One example of the mixed testing signals

Table 2. Crosscorrelation coefficients for speech signals (testing)

comparable parameters / names of decorrelator	Crosscorrelation coefficient for mixing instantly	Crosscorrelation coefficient for mixing in time
Multi-FIR	0.0054 (Order 3)	0.1254 (Order 6)
FIR	0.0058 (Order 3)	0.1477 (Order 6)
Gamma	0.0061 (Order 2)	0.1495 (Order 5)
Nonlinear FIR	0.0054 (Order 3)	0.1474 (Order 6)
Scalar	0.0064	0.1934

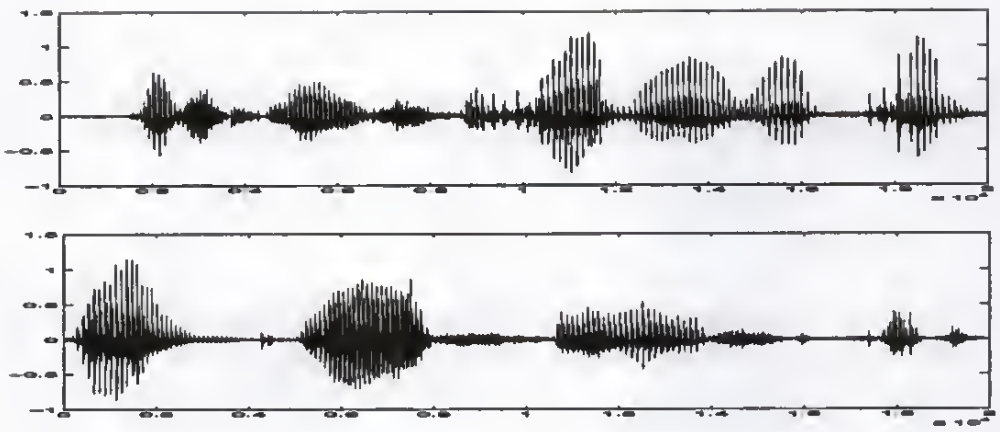


Figure 56. Reconstructed signals from testing

From both Table 1 and Table 2, it can be concluded that the Multi-FIR learning produces the best decorrelator. Another important observation is that instantaneous mixing produces less correlation than mixing in time, although the corresponding training results are comparable. This reflects the fact that decorrelation for mixing in time is a much harder problem than for mixing instantly. But we should point out that the testing results for mixing in time are also of good quality since the interference is almost totally cancelled, as can be visually appreciated by comparing the results given in Figure 48 and Figure 56. Moreover, we played the speech through a loud found that the interference is almost unnoticeable.

With multi-Gamma, multi-nonlinear filter, similar conclusions can be obtained. Generally speaking, the multi-decorrelators such as multi-FIR, multi-Gamma, and multi-nonlinear filter are able to give better decorrelation results for nonstationary signals since much more nonstationary information is utilized in the decorrelation.

Now, we will show that the proposed method also works for stationary Gaussian signals. This is important point since the algorithms based on higher-order statistics do not work for Gaussian signals since their cumulants are zero. Two uncorrelated stationary

Gaussian random processes are used as original sources. The mixed signals are generated by mixing instantly and in time. The learning curve of multi-FIR for mixing them instantly is shown in Figure 57. For comparison, we also provide the FIR learning curve with the same order as the multi-FIR. And for the mixing in time case, learning curves for both multi-FIR and FIR are shown in Figure 58. The final crosscorrelation coefficient of the separated signals are presented in Table 3. It can be seen from the final crosscorrelation coefficients that the multi-FIR is better than the FIR in both mixing instantly and mixing in time, since even for stationary signals the full crosscorrelation matrix is needed.

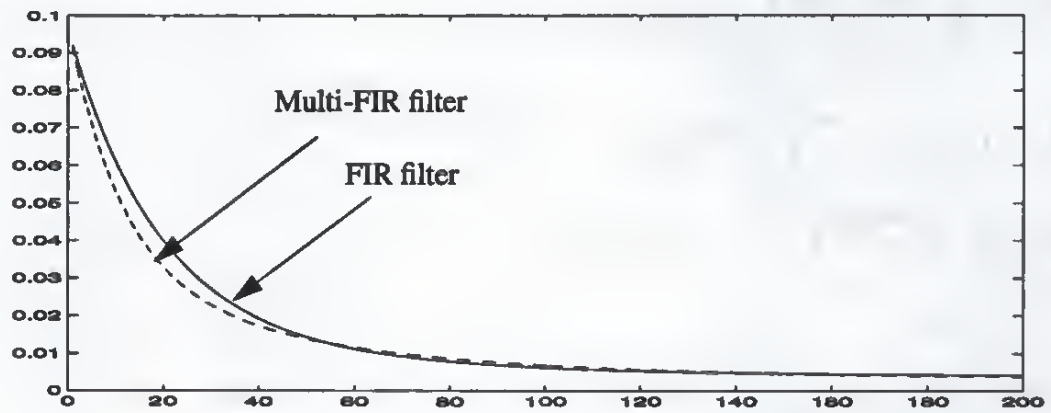


Figure 57. The learning curves of multi-FIR and FIR filters for stationary Gaussian signals mixing instantly

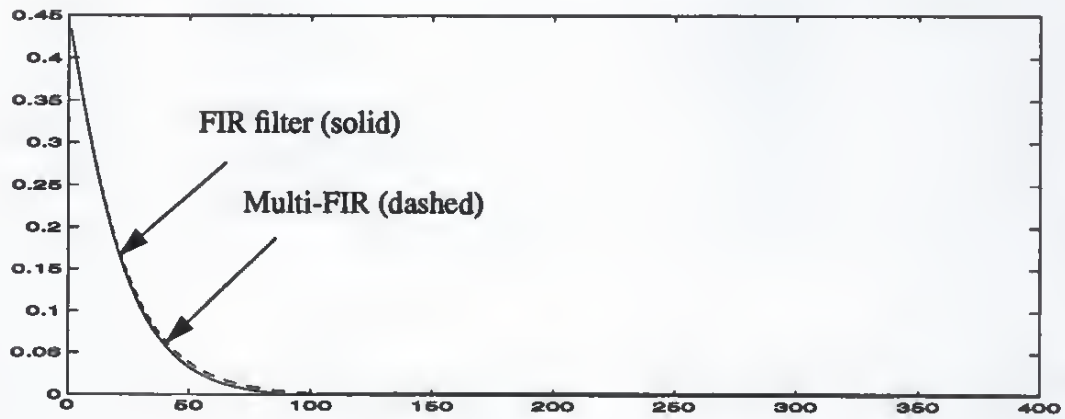


Figure 58. The learning curves of multi-FIR and FIR filters for stationary Gaussian signals mixing in time

Table 3. Crosscorrelation coefficients for separated Gaussian signals(training)

comparable parameters / names of decorrelator	Final crosscorrelation coefficient for mixing instantly	Final crosscorrelation coefficient for mixing in time
Multi-FIR	3.4×10^{-3} (Order 3)	1.7×10^{-3} (Order 6)
FIR	3.9×10^{-3} (Order 3)	2.2×10^{-3} (Order 6)

Figure 59. One of the training speech signals corresponding to the minimum point of learning curve for multi-FIR case shown in Figure 53

From the learning curves shown in Figures 53, 54, 57, and 58, it is known that there are several unstable extreme points existing for nonstationary signals, although the learning procedure stays at a stable point finally; for stationary signals, however, the learning procedure goes to the unique stable point. We tried to keep the learning stay in those points with very small coefficients for nonstationary signals, but the training results are worse. One example, which is from the multi-FIR case, is shown in Figure 58. Of course, the test results are also worse. It is very difficult to analyze the performance of a filter due to nonstationarity of the signals, we just point out this observation here as a further research direction.

4.4 Discussions

As an application of the theory studied in this thesis, we presented a learning network to estimate the temporal crosscorrelation between two signals and utilized it in a

neural topology for source separation. The new network is much simpler than other neural approaches, and is able to compute the temporal crosscorrelation function for either stationary or nonstationary signals. Based on it, we showed a new approach to separate blind mixed signals. This approach outperforms others because it provides enough information for both stationary and nonstationary signals and can be used either for instantly mixing or mixing in time, and with Gaussian or non-Gaussian signals.

CHAPTER 5

AN INFORMATION-THEORETIC PERSPECTIVE FOR LEARNING SYSTEMS

5.1. Introduction

In Chapter 2, we showed that most supervised and unsupervised learning systems employ the correlation idea in the adaptation their rules, but the relationship between supervised and unsupervised are still not revealed. The goal of this chapter is to provide a unified perspective for supervised and unsupervised learning systems.

As mentioned in Chapter 1, the conventional distinction between these two kinds of learning system is whether a teacher signal is used in learning. In learning with supervision, it is traditionally assumed that at each time instant we know in advance the desired response for the learning system, and we use the difference between the desired and the actual response, that is, the error, to correct its behavior [Tsypkin, 1971]. In unsupervised learning, the desired response of the learning system is not explicitly known, and thus we cannot directly formulate and use the error of the learning system in order to improve its behavior. In this chapter, we will show that the distinction between supervised and unsupervised learning is not in the existence of a desired signal, but whether there is common information between the input and desired signals. We will concentrate on "when supervised learning degenerates to unsupervised learning?", since the supervised system is "more encompassing" than unsupervised learning due to the existence of the extra desired signal. For supervised learning, the MSE based algorithm is considered since it is the most popular error criterion used in neural networks. Two cases will be investigated: The first

one is that the input and the desired signals are the same, which is referred to autoassociative supervised learning; and the second one discusses the case when the input and the desired signals are different, which is referred to heteroassociative supervised learning.

This chapter is organized as follows: We will study the autoassociative supervised learning in section 5.2. And in section 5.3 the heteroassociative supervised learning will be dealt with. Simulation results are given in section 5.4. Some necessary elements for information theory will be outlined in section 5.5. The main results of this chapter is summarized in section 5.6. Discussions are given in section 5.7.

It should be pointed out that some of content of this chapter overlaps with chapter 2 due to the following two considerations:

(1) This can be viewed as a deep step of the results of Chapter 2.

(2) It will be easy to understand the discussions given in this chapter if the chapter is organized in a relatively independent way.

5.2. Relationship Between the Autoassociative Supervised Learning and Unsupervised Learning

The learning system to be considered in this section is the one shown in Figure 60, but with a constraint $x(t) = d(t)$. We will illustrate that the autoassociative supervised learning is equivalent to an unsupervised learning and provide some examples.

5.2.1 Autoassociation by Multilayer Perceptron

It has been shown mathematically that a linear or nonlinear multilayer perceptron can be used to extract principal components information of the input signal if the desired signal is the input itself and the error criterion is the MSE as given by

$$J = \frac{1}{2}E\left[\sum_i (x_i(t) - y_i(t))^2\right] \tag{131}$$

where $y_i(t)$ is the output of the network [Baldi and Hornik, 1989; Bourlard and Kamp, 1988]. Bourlard and Kamp proved that the nonlinearities of the hidden units are useless and that the optimal parameters can be derived by purely linear techniques relying on PCA; that is, the autoassociation MLP is equivalent to a linear single layer network trained with normalized Hebbian rule.

5.2.2 Linear, Nonlinear PCA Networks

Oja and Karhunen studied the details of how to use a linear and nonlinear network to extract principal components information of a signal x(t) [Oja, 1992; Oja and Karhunen, 1993]. Let us assume that, without loss generality, a single layer network with nonlinear neurons, the mapping function of the network can be described by $y = G(W^T x)$, where y is the output and T is the transpose operator, W is the weight vector of the network, and x is the input vector. It is shown that minimizing the MSE cost function given by Eq. (131) followed by a normalization on weight vector the outputs of the network represent either the linear PCA when G is a linear function, or the nonlinear PCA when G is a nonlinear function.

$$J = \frac{1}{2}E\left[(x - WG(W^T x)^2)\right] \tag{132}$$

Moreover, Karhunen and Joutsensalo also showed that the corresponding linear and nonlinear PCA can be implemented by constrained Hebbian rule in unsupervised modes [Karhunen and Joutsensalo, 1994, 1995].

From the above two typical examples, we see that the autoassociative supervised

learning degenerates to unsupervised learning with the constraint $x(t) = d(t)$. This constraint both the input and the desired signal have the same probability distribution function and are fully statistical correlated.

5.3. Relationship Between the Heteroassociative Supervised Learning and Unsupervised Learning

In this section, we study supervised learning under the constraint $x(t) \neq d(t)$, and try to figure out when the so called heteroassociative supervised learning degenerates to unsupervised learning.

5.3.1 Heteroassociative Supervised Learning in Linear Systems

First, let's consider the supervised learning with the MSE in a linear network depicted in Figure 60.

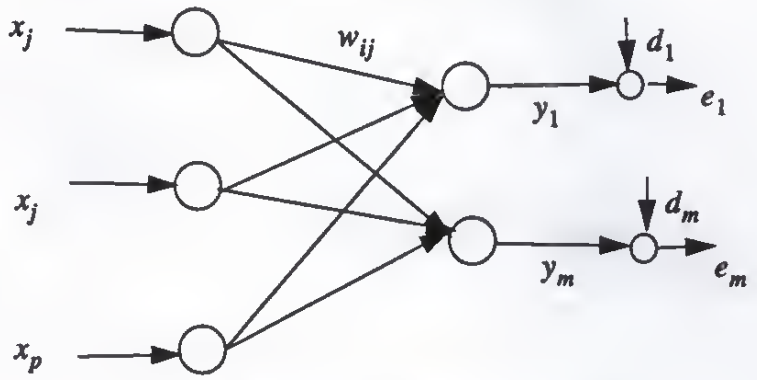


Figure 60. A linear network

where, j=1,2,..., p; i=1,2,..., m, and the desired signals $d_i(t)$ and error signals $e_i(t)$ are used in supervised learning.

The outputs of the network are

$$y_i(t) = \sum_j w_{ij}(t)\, x_j(t) \tag{133}$$

where w_{ij} is the weight connecting the input x_j to the output y_i.

The off-line adaptation rule is

$$\begin{aligned} \Delta w_{ij}(t) &= -\eta \nabla_{w_{ij}} J = \eta E\left[(d_i(t) - y_i(t))\, x_j(t) \right] \\ &= -\eta E\left[y_i(t)\, x_j(t) \right] + \eta E\left[d_i(t)\, x_j(t) \right] \end{aligned} \tag{134}$$

where ∇ represents the gradient operator.

The corresponding on-line learning algorithm has the same form as Eq. (134), except that it drops the statistical expectation operator.

Notice that the first term in the right hand side (RHS) of Eq. (134) is the statistical version of the anti-Hebbian rule and the second term in the RHS of Eq. (134) is the forced Hebbian rule. Forced Hebbian is the rule which substitutes the desired response into the Hebbian learning instead of the output, so it is nothing but correlation learning. Therefore, in a linear network, learning with the MSE is statistical equivalent to learning with the combination of the forced Hebbian and the anti-Hebbian rule.

In order to bridge the connection between the heteroassociative supervised learning and unsupervised learning, we consider the case of random noise as the desired signal. Intuitively this should lead to some form of unsupervised learning since the desired signal is random noise which does not contain any information.

Proposition 1:In a linear network, if the desired signal is a zero mean random sequence independent of $x_j(t)$, or $d_i(t)$ and $x_j(t)$ are orthogonal, the forced Hebbian in the right hand side of Eq. (134) becomes zero, and then the MSE learning defaults to anti-Hebbian. That is, we have

$$\Delta w_{ij}(t) = -\eta E\left(y_i(t)\, x_j(t) \right) \tag{135}$$

An on-line version of Eq. (135) is exactly the anti-Hebbian rule. Therefore, we conclude that supervised learning with the MSE in a linear network is the same as unsupervised learning with anti-Hebbian rule if the desired signal is a zero mean random noise or it is orthogonal to the input signal.

It is interesting to note that when the desired signal is orthogonal to the input signal, it is also orthogonal to the output signal since the output is a linear combination of input signals. Geometrically, this relation can be depicted in Figure 61. The learning procedure becomes the search of a proper output y in the input space guided by the anti-Hebbian term. It is obvious that the desired signal has no effect on the output and the learning procedure is self-learning.

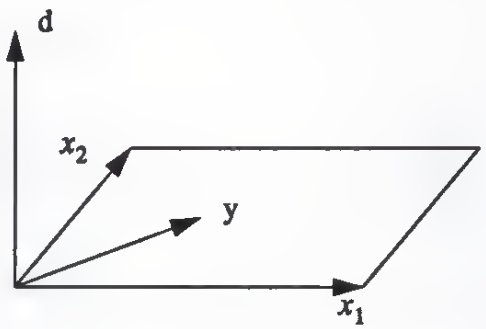


Figure 61. The learning procedure of linear system

Another look at the relation between unsupervised and supervised learning can be given by the minimizing output energy.

Proposition 2: The anti-Hebbian rule is equivalent to MSE learning when the desired signal is a zero mean random noise. We will prove this proposition by minimizing the output energy function.

Proof: rewritten Eq. (131)

$$J = \frac{1}{2}E(\sum_i (d_i(t) - y_i(t))^2) \tag{136}$$

It is not difficult to show [Richard, Lippmann 1991] that Eq. (136) is equal to

$$J = \frac{1}{2}E\{\sum_i (y_i(t) - E\{(d_i(t)) \mid (x(t))\})^2\} + E\{\sum_i var((y_i(t)) \mid (x(t)))\} \tag{137}$$

Where '|' represents 'conditional'.

When the desired signal d_i is a random noise with zero mean, $E\{d_i|x\} = 0$. And the second term in Eq. (137) is independent of the weights, so minimizing Eq. (137) is equivalent to minimizing the output energy function

$$J = \frac{1}{2}E\{\sum_i (y_i)^2\} \tag{138}$$

Minimizing Eq. (138) in a linear network can be accomplished by a stochastic gradient-descent search with the anti-Hebbian rule [Palmieri et al. 1993].

$$w_{ij}(t+1) = w_{ij}(t) - \eta y_i(t) x_j(t) \tag{139}$$

It is interesting to note that when the step size in MSE is set to a negative value($-\eta$), the optimization process goes along the gradient ascent direction, i.e., the unsupervised rule becomes the basic Hebbian rule:

$$\Delta w_{ij}(t) = \eta E[y_i(t) \times x_j(t)] \tag{140}$$

In this situation, the optimization is unstable, which is consistent with the known instability of the Hebbian rule in the sense that network coefficients will grow without bound. From the system energy point of view, we are maximizing the system output energy. So, when the step size is negative, MSE has similar properties to the Hebbian

learning.

The results obtained above have obviously geometrical means since all variables and relationships are discussed in a linear space. We are going to extend the above results to the nonlinear system, which will be presented in subsequent section.

5.3.2 Heteroassociative Supervised Learning in Nonlinear Systems

It is well known that nonlinearity is a very important characteristic of learning systems, especially for neural networks. With nonlinearity, the performance of the whole system can be improved greatly. Hence, it is also very significant to extend the results obtained in section 5.3 to nonlinear systems.

For a nonlinear learning system, the output has the form

$$y_i(t) = f(w(t), x(t)) \tag{141}$$

where f is the mapping function between the input and output of the nonlinear system. And the adaptation rule with respect to MSE becomes

$$\begin{aligned} \Delta w_{ij}(t) &= -\eta \nabla_{w_{ij}} J = \eta E[(d_i(t) - y_i(t)) f'_{w_{ij}}(t)] = \\ & -\eta E[y_i(t) \times f'_{w_{ij}}(t)] + \eta E[d_i(t) \times f'_{w_{ij}}(t)] \end{aligned} \tag{142}$$

where $f'_{w_{ij}}(t)$ is the differential of f with respect to the weight w_{ij}.

If the desired signal is a random sequence independent of $f_{w_{ij}}(t)$, or $d_i(t)$ and $f'_{w_{ij}}(t)$ are orthogonal, the second term in the RHS of Eq. (142) is zero and we have a similar result to the linear case:

$$\Delta w_i(t) = -\eta E[y_i(t) \times f'_{w_{ij}}(t)] \tag{143}$$

Under this hypothesis, we need the desired signal to be independent of $\dot{f}_{w_{ij}}(t)$. It is not difficult to show the validation of Eq. (143), based on the following theorem, when the desired signal is a zero mean random noise.

Theorem (Feller, 1966): If random variables ζ and ϑ are independent of each other, and the functions g and f are Borel functions, then, $f(\zeta)$ and $g(\vartheta)$ are independent too.

Most real functions f(x) including the sigmoid function used extensively in neural networks, which are random functions when the x is a random variable, are Borel functions. Therefore, satisfies the condition of the above theorem. Hence, we can conclude that when the desired signal is a zero mean white noise, $\dot{f}_{w_{ij}}(t)$ and $d_i(t)$ are independent of each other.

We are showing the details of Eq. (143), as an example, using a two-layer feedforward neural networks which is shown in Figure 62.

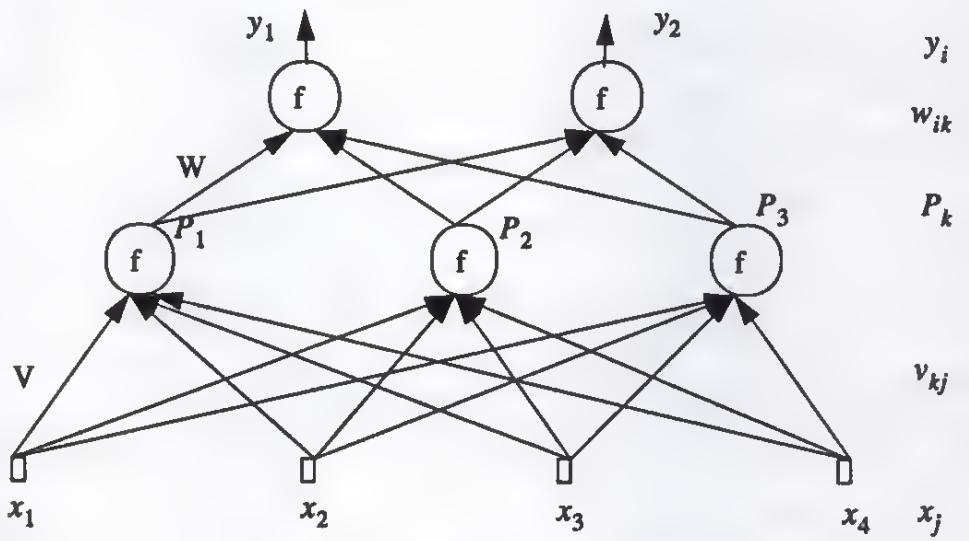


Figure 62. A two layers MLP

It should be noted that in nonlinear network the weight adjustment is a linear process if gradient descent based back-propagation is used as a optimization strategy [Principe, 1996]. The feedforward network which is a nonlinear network is shown in Figure 62, and

the backward network which is a linear network with the dual structure of the feedforward network can be depicted in Figure 62'. This point is very important since in linear systems, both feedforward and dual networks are linear, while in the nonlinear case this not true.

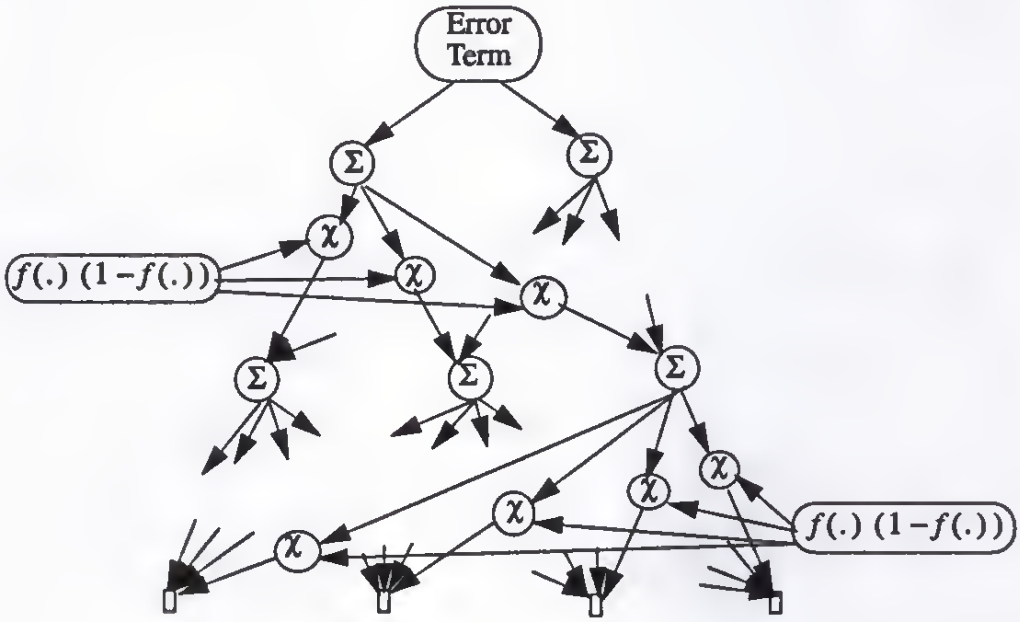


Figure 62'. The backward network corresponding the network in Figure 62

In the figures, the χ represents the multiply operator, and $f(.)\ (1-f(.))$ operator share the same form for all weights but with different variables.

Let us now derive the learning in this network [Hertz et al., 1991]. Given Pattern α, hidden unit k receives a net input

$$h_k^\alpha = \sum_j V_{kj} x_j^\alpha \tag{144}$$

and produces output (activation of the hidden layer)

$$P_k^\alpha = f(h_k^\alpha) = f(\sum_j V_{kj} x_j^\alpha) \tag{145}$$

Output unit i thus receives

$$h_i^\alpha = \sum_k W_{ik} P_k^\alpha = \sum_k W_{ik} f(\sum_j V_{kj} x_j^\alpha) \tag{146}$$

and produces for the final output

$$y_i^\alpha = f(h_i^\alpha) = f(\sum_k W_{ik} P_k^\alpha) = f(\sum_k W_{ik} f(\sum_j V_{kj} x_j^\alpha)) \tag{147}$$

Then, using the MSE criterion given by Eq. (131) and gradient decent method, we get

(1) Weights between the output layer and hidden layer are adjusted in the following way

$$\Delta W_{ik} = \eta E[\varsigma_i^\alpha P_k^\alpha] \tag{148}$$

where we have defined

$$\varsigma_i^\alpha = f'(h_i^\alpha)[d_i(t) - y_i(t)] \tag{149}$$

(2) Weights of other layers are adjusted by

$$\Delta V_{kj} = \eta E[\varsigma_k^\alpha x_j^\alpha] \tag{150}$$

with

$$\varsigma_k^\alpha = f'(h_k^\alpha) \sum_i W_{ik} \varsigma_i^\alpha \tag{151}$$

where the expectation operator is done with respect to all patterns.

Decomposing Eq. (148) into

$$\begin{aligned}\Delta W_{ik} &= \eta E[\varsigma_i^{\alpha} P_k^{\alpha}] = \eta E[f'(h_i^{\alpha})\,[d_i(t) - y_i(t)]\,P_k^{\alpha}] \\ &= \eta E[f'(h_i^{\alpha})\,d_i(t)\,P_k^{\alpha}] - \eta E[f'(h_i^{\alpha})\,y_i(t)\,P_k^{\alpha}]\end{aligned} \tag{152}$$

Similarly, Eq. (150) can be decomposed into

$$\begin{aligned}\Delta V_{kj} &= \eta E[\varsigma_k^{\alpha} x_j^{\alpha}] = \eta E\Big[\big(f'(h_k^{\alpha})\sum_i W_{ik}\varsigma_i^{\alpha}\big)\,x_j^{\alpha}\Big] \\ &= \eta E\Big[\big(f'(h_k^{\alpha})\sum_i W_{ik}\big(f'(h_i^{\alpha})\,[d_i(t) - y_i(t)]\big)\big)\,x_j^{\alpha}\Big] \\ &= \eta\sum_i \{E[f'(h_k^{\alpha})\,W_{ik}f'(h_i^{\alpha})\,[d_i(t) - y_i(t)]\,x_j^{\alpha}]\} \\ &\eta\sum_i \{E[f'(h_k^{\alpha})\,W_{ik}f'(h_i^{\alpha})\,d_i(t)\,x_j^{\alpha}]\} - \eta\sum_i \{E[f'(h_k^{\alpha})\,W_{ik}f'(h_i^{\alpha})\,y_i(t)\,x_j^{\alpha}]\}\end{aligned} \tag{153}$$

where the expectation operator and summation have been exchanged.

Since we assume that the desired signal $d_i(t)$ is a zero mean white noise, and the nonlinear function f in the MLP is a Borel function, we known that $d_i(t)$ is independent of P_k and any other function which is a function of the input x. Hence, Eq. (152) can be written into

$$\Delta W_{ik} = -\eta E[f'(h_i^{\alpha})\,y_i(t)\,P_k^{\alpha}] \tag{154}$$

Similarity, Eq. (153) can be simplified into

$$\Delta V_{kj} = -\eta\sum_i \{E[f'(h_k^{\alpha})\,W_{ik}f'(h_i^{\alpha})\,y_i(t)\,x_j^{\alpha}]\} \tag{155}$$

If we define two new variables as

$$\Theta_i^{\alpha} = f'(h_i^{\alpha})\, y_i(t) \tag{156}$$

and

$$\Theta_k^{\alpha} = f'(h_k^{\alpha}) \sum_i W_{ik} \Theta_i^{\alpha} \tag{157}$$

which are the signal back-propagated to the output neurons and the hidden neurons, respectively. Then, Eqs. (154) and (155) can be rewritten into

$$\Delta W_{ik} = -\eta E[\Theta_i^{\alpha} P_k^{\alpha}] \tag{158}$$

and

$$\Delta V_{kj} = -\eta E[\Theta_k^{\alpha} x_j^{\alpha}] \tag{159}$$

From Eqs. (158) and (159), it is clear that all weights in the network are adapted based on the nonlinear anti-Hebbian rule. Although the adjustments include the input variables for the weights between input and hidden layer and the activation variables for the between the output and hidden layer, the other variables are not exact by the outputs of the corresponding layer for the input and the outputs of corresponding layer for the activation, respectively. Actually, the adjustments are consistent with the variables in the dual network which is resulting from gradient operator and thus is a linear network.

Although similar results can be obtained in the nonlinear systems, it is not easy to declare the relationships geometrically in nonlinear manifold.

So far, we have shown that the heteroassociative supervised learning degenerates to unsupervised anti-Hebbian learning when the desired signal is a zero mean random noise

which is statistical independent of the input signal. Next, some experimental results will be reported to verify the analysis given above.

5.4. Simulation Results

In order to confirm the theoretical analysis given above, we provide some computer simulation results in this section. First of all, we analyze the anti-Hebbian learning (we do the experiments with anti-Hebbian rule instead of Hebbian rule due to the instability of Hebbian rule) in a linear network for understanding better the equivalence between the supervised and the unsupervised learning.

5.4.1 Anti-Hebbian Learning and its Representation in Linear Signal Space

The basic linear network for unsupervised learning is shown in Figure 63.

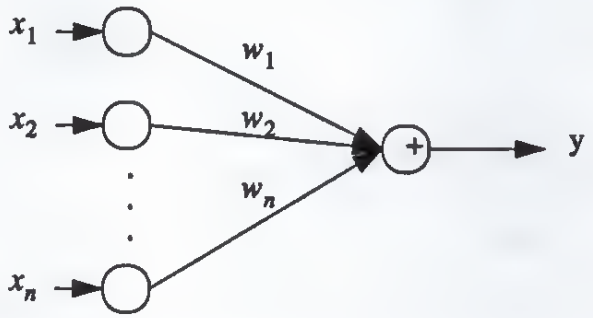


Figure 63. A linear network with one output unit

Let $x = [x_1, x_2, \ldots, x_n]$ be the input vector, $w = [w_1, w_2, \ldots, w_n]$ be the weight vector, and the scalar y be the output.

The vector representation of the anti-Hebbian learning has the form of

$$\Delta w(n) = -\eta y(n) x(n) \tag{160}$$

And in a linear network, we know the output

$$y(n) = w(n) x(n)^T \tag{161}$$

where T denotes the transpose operator.

Eq. (159) tells us that $\Delta w(n)$ is the outer product of the output and the input vector. When the output y is a scalar, each $\Delta w(n)$ is parallel to the input space. Hence, the cumulative weight vector increment w_{incre} which is defined as the difference between the final weight and the initial vectors is also parallel to the input space. We can describe the relationship among vectors in the anti-Hebbian learning in a signal space as given in Figure 64,

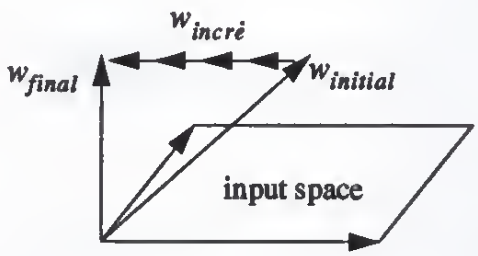


Figure 64. The signal space description of the anti-Hebbian learning

where $w_{initial}$ represents the initial weight vector, w_{final} is the weight vector after training, and w_{incre} is the cumulative increment of the weight vector during adaptation.

The cumulative of weight vector is a very important variable since it represents the new information the network learns from the training data set with a given learning rule.

It is very useful to notice in the anti-Hebbian learning that the w_{incre} is parallel to the input space in each learning step and the final weight vector w_{final} is just the orthogo-

nal projection of the initial weight vector onto orthogonal complement of the input space. Hence, the anti-Hebbian learning in a linear network is a process to find the projection of the initial weight vector onto the orthogonal complement of the input space.

5.4.2 Experimental Results in the Linear System

The network used is depicted in Figure 63. The input signal is a sinusoid with frequency 1/50 Hz which is shown in Figure 65. The length of the segment data is 50. And the desired signal is a zero mean white noise which is shown in Figure 66. In the simulation, the same network architecture is applied for both supervised and unsupervised learning, except that a desired signal is employed for supervised learning and but not for unsupervised learning.

We do the experiment 20 times in order to see the statistical behavior with the same input signal and different initial weight vector which is set randomly. The cumulative of weight vector is shown in Figure 67 for the unsupervised anti-Hebbian learning and in Figure 68 for the supervised learning using zero mean random noise as the desired signal.

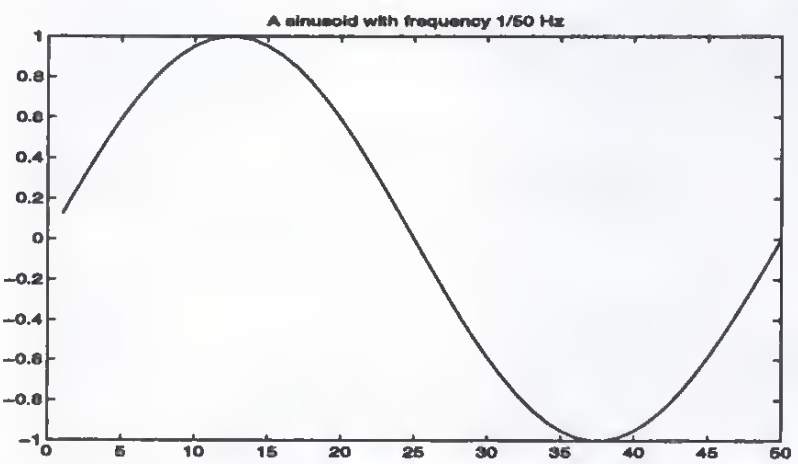


Figure 65. The input signal

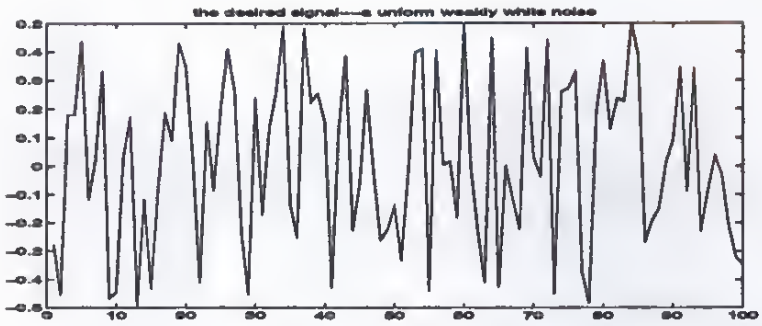


Figure 66. The desired signal

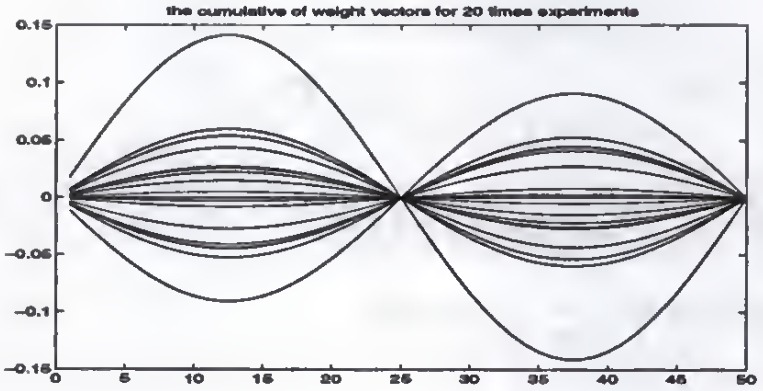


Figure 67. Results for unsupervised learning

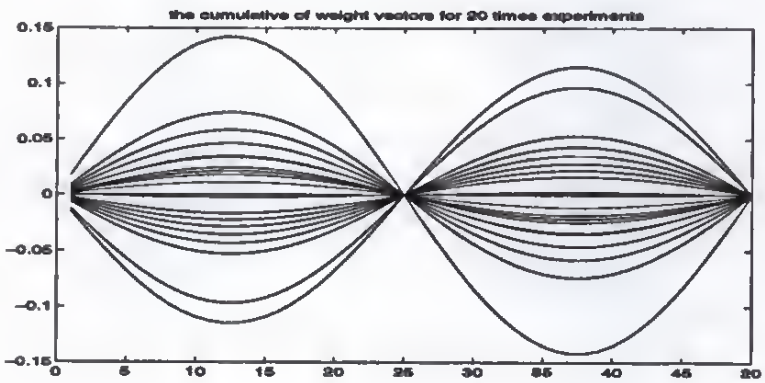


Figure 68. Results for supervised learning

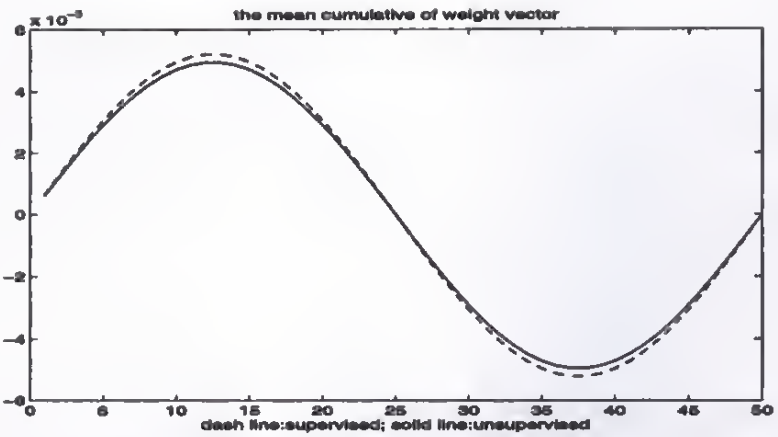


Figure 69. The mean of cumulative of weight vectors

The results given in Figures 67 and 68 are two sinusoid-like waves and are parallel to the input signal in Figure 66 since they have the same phase. This is what the outcomes of section 5.1 tell us.

The mean values of the cumulative of weight vectors for the 20 times experiments are given in Figure 69. It is obvious that from the statistics point of view the cumulative of weight vectors are the same in both learning paradigms. Therefore, we can conclude from the simulation results that the supervised learning changes qualitatively into the unsupervised learning when the desired signal is a zero mean random noise.

5.4.3 Results from Nonlinear System

We are now doing simulations for a two-layer multilayer perceptron (MLP) given in Figure 62. In order to express the results clear, we redraw and relabel the variables as in Figure 70. The input signal is also a sine wave but with different length for suiting the network, which is shown in Figure 71. The desired signal is the same as for the linear network.

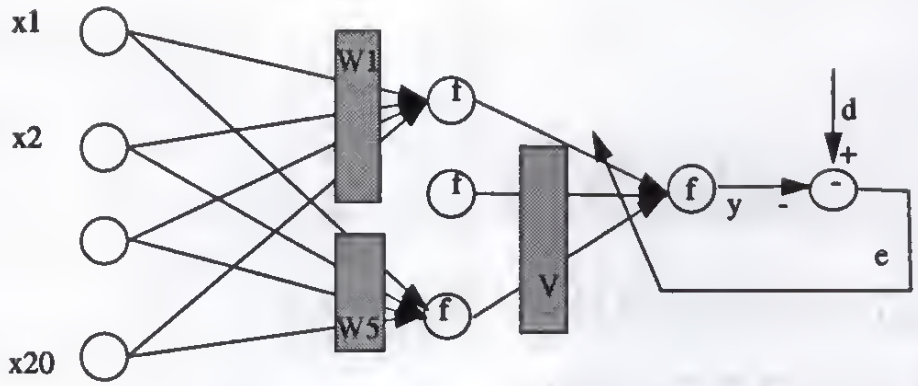


Figure 70. Two-layer nonlinear neural network

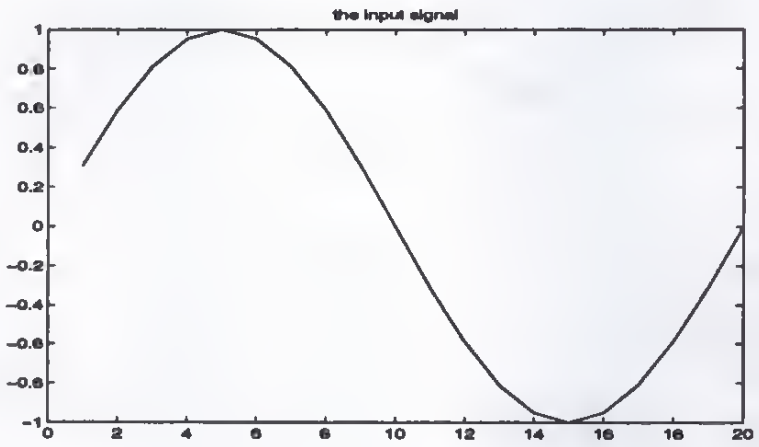


Figure 71. The input signal

In the network, we are labelling the weight vectors as W1, W2, W3, W4, and W5 represent the weights connecting to the first neuron, the second neuron, and so on in the hidden layer, that is, the length of each weight vector is 20. And the weights between the hidden and the output layer are labelled V.

It should be pointed out that when the nonlinear anti-Hebbian rule is used here to train the nonlinear network the coefficients $f(h_i^{\alpha})$ and $f(h_k^{\alpha})$ given by Eqs. (145) and (147) should be multiplied to the corresponding learning rules.

Figure 72 shows the cumulative increment (C-I) of weight vectors after 1000 itera-

tions for one experiment in supervised learning.

It is clear that the C-I of each weight vector looks like a sine wave since learning with anti-Hebbian decreases the amplitude of the output which means that the nonlinearity works in the near linear region, such that the C-I of weight vectors almost follows the input signal. The waveform of C-I of V is hard to be identified since it is related to the activations of the hidden layer which are the results of nonlinear anti-Hebbian learning.

Comparing with the C-I of weight vectors, which are shown in Figure 73, of unsupervised anti-Hebbian learning, we can see both of them have similar characteristics. The sign of sin wave is decided by the initial weights.

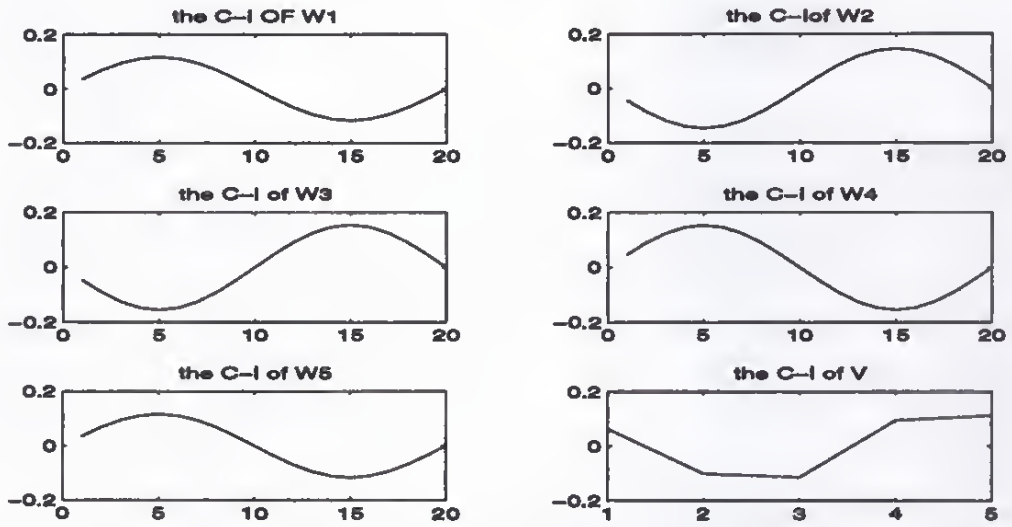


Figure 72. The C-I for each weight vector in on experiment for supervised learning

Now, let us see the statistical characteristics of C-I of weight vectors. The 20 times C-I of W1 are shown in Figure 74. We only present the C-I of W1 since the C-I of other weight vectors have similar properties.

Figure 75 gives us the 20 times C-I for the second layer. Figure 76 is the mean value of all weight

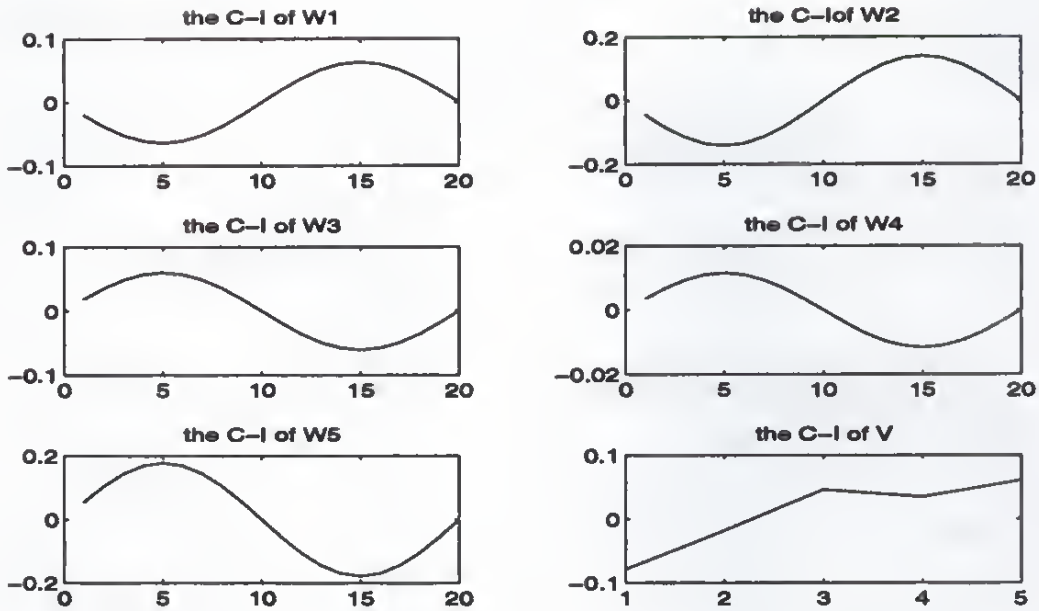


Figure 73. The C-I for each weight vector in on experiment for unsupervised learning

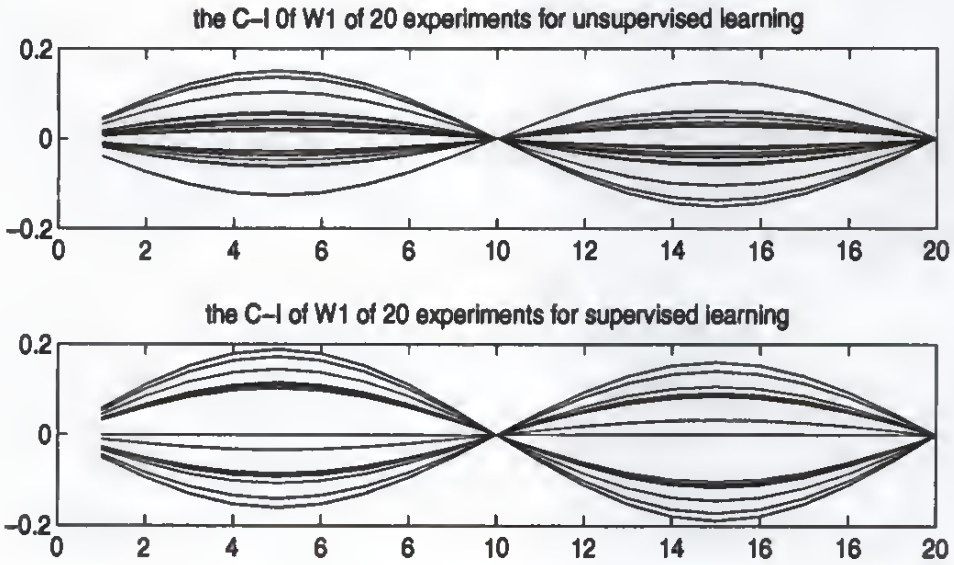
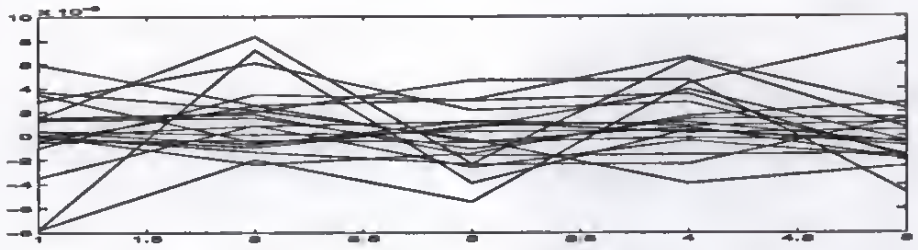


Figure 74. The C-I for 20 experiments

The C-I of V for 20 experiments for unsupervised learning



The C-I of V for 20 experiments for supervised learning

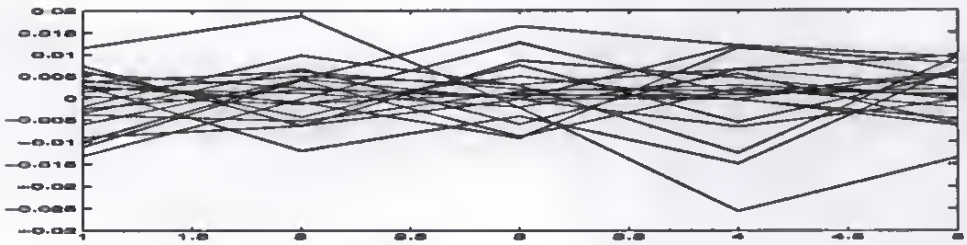


Figure 75. The C-I of V for 20 experiments

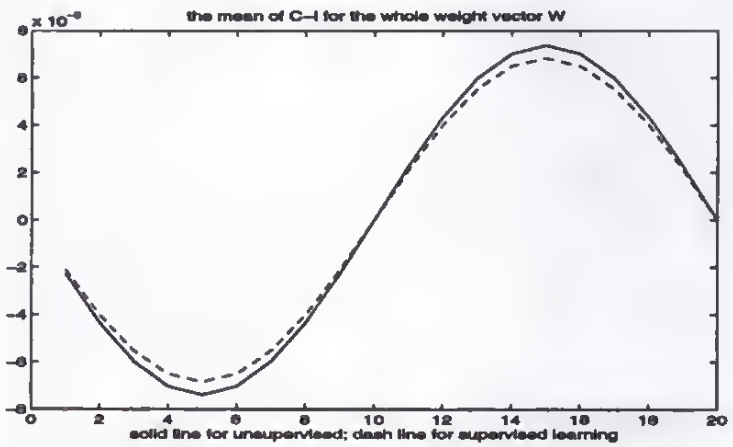


Figure 76. The mean of C-I for the weight vector W

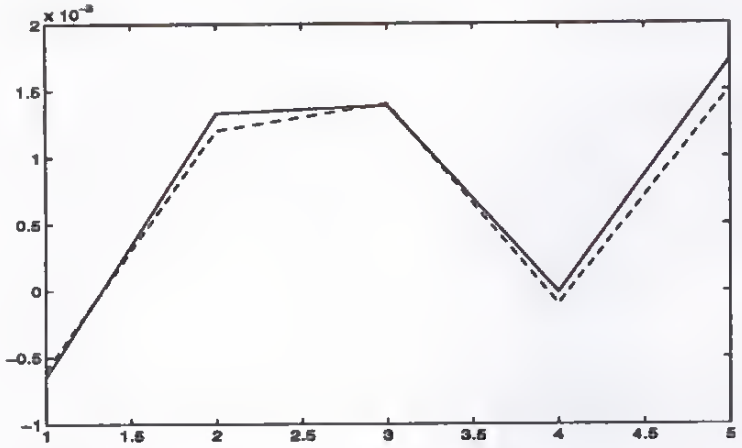


Figure 77. The C-I of V for 20 experiments

vector W=[W1 W2 W3 W4 W5] for 20 time experiments. It tells us the supervised and unsupervised learning have the same C-I of weight vector of the first layer. Figure 77 presents the mean value of the weight vector V in the second layer, from which it is clear the means are very close since they are in the same region, although they are not exactly the same. It should be mentioned that the learning rate for unsupervised system is much faster than supervised systems.

The results obtained in this section can be summarized as follows: we have shown theoretically and experimentally that the heteroassociative supervised learning can degenerate to unsupervised learning when the desired signal is statistically independent of the input signal. That is, the desired signal and the input do not correlate at all. This conclusion combined with the conclusion obtained in section 5.2 will be enhanced in next section from the information point of view.

5.5 Basic Elements of Information Theory

In this section, we review the basic concepts of information theory which are useful

to better understanding the contents in this section without proof. For more details, the reader s are referred to any text on information theory such as [Jumarie 1990] or [Kapur and Kesavan, 1992].

(1) Entropy:

Let's first look at Shannon's entropy. Suppose we have a set of possible events A whose probability of occurrence are $p_1, p_2, \ldots, p_n$. These probabilities are known but that is all we know concerning which event will occur. Shannon defined the information of this sequence as the so-called entropy [Shannon 1949]:

$$H = -\sum_i p_i \ln p_i \tag{162}$$

It is well known that a source entropy gets its maximum value, which is a constant, when p_i has a uniform distribution. Entropy is measurement of uncertainty of a source signal. It is also important to note that the information between two sources, which is illustrated by the concept of

mutual information:

(2) Conditional Entropy:

Consider two random experiments $\alpha(p_1, p_2, \ldots, p_m)$ for event A and $\beta(q_1, q_2, \ldots, q_n)$ for event B which are not necessary statistical independent; in other word, the equation

$p(A_i B_j) = p(A_i)\, p(B_j)$ does not necessary hold, but rather, one has:

$$p(A_i B_j) = p(A_i)\, p(B_j | A_i) \qquad \forall i \quad \forall i \tag{163}$$

where $p(\beta_j | \alpha_i)$ denotes the conditional probability of β_j given α_i.

Definition of conditional entropy: The conditional entropy $H(\beta | \alpha)$ of the random experiment β given the random experiment α is defined as

$$H(\beta|\alpha) = \sum_{i}^{m} p(A_i) H(\beta|A_i) \tag{164}$$

where $H(\beta|A_i)$ is given by the expression

$$H(\beta|A_i) = -\sum_{j} p(B_j|A_i) \ln p(A_i B_j) \tag{165}$$

It is easy to show from Eqs. (163), (164) and (165) that the following equation holds:

$$H(\alpha\beta) = H(\alpha) + H(\beta|\alpha) \tag{166}$$

$H(\beta)$ is a measure of the entire uncertain involved in β, irrespective of any external reference; $H(\beta|\alpha)$ measures the uncertainty concerning β given that we previously performed the experiment α.

(3) Mutual Information

Definition of mutual information: Consider two random experiments $\alpha(p_1, p_2, \ldots, p_m)$ for event A and $\beta(q_1, q_2, \ldots, q_n)$ for event B, the mutual information $I(\beta|\alpha)$ between α and β is defined by the expression:

$$I(\beta|\alpha) = H(\beta) - H(\beta|\alpha) \tag{167}$$

Eq. (167) represents the amount of information contributed by α about β, or likewise amount of information contained in α about β. And it can also be represented in probability form,

$$I(\beta|\alpha) = E\left(\log \frac{p(A_i B_j)}{p(A_i)\, p(B_j)}\right) \tag{168}$$

Mutual information is also called the transinformation which means the amount of information contained in α about β.

It is very important to understand that the entropy of (122) is a measure of uncertainty, while the information is a difference of uncertainties, that is to say a difference of entropies [Jumarie 1990]. Indeed, several scientists consider the entropy $H(\alpha)$ as an amount of information, and strictly speaking this is not wrong. However, it is deeply misleading for the very reason that $H(\alpha)$ is the measure of an amount of information in some special cases. Basically $H(\alpha)$ defines a degree of uncertainty, and incidentally only this is equivalent to an information. This unfortunate habit originated from the equation

$$I(\alpha|\beta) = H(\alpha) - H(\alpha|\beta) \tag{169}$$

in which $I(\alpha|\beta)$ is an amount of information, while $H(\alpha)$ and $H(\alpha|\beta)$ measure the respective uncertainties involved in α before and after making the experiment β. Incidentally, when $H(\alpha|\beta)$ =0, that is to say when β does not contain any information about α, then $I(\alpha|\beta)$ =$H(\alpha)$.

Information is no more than a difference of uncertainties, and in the following we shall carefully utilize "uncertainty" and "information" according to their respective meanings.

Some properties of mutual information:

(1) The mutual in non-negative.

$$I(\alpha|\beta) \geq 0 \tag{170}$$

The equality holds if and only if

$$H(\alpha|\beta) = H(\alpha) \tag{171}$$

(2) The mutual information is symmetric,

$$I(\alpha|\beta) = I(\beta|\alpha) \tag{172}$$

(3) The maximum amount of information involved in α about β is $H(\alpha)$ and the limit is achieved if and only if $H(\alpha|\beta)$ =0, that is to say, when α is completely determined by β.

5. 6 An Information-Theoretic Perspective for Learning Systems

We provide a unified perspective based on mutual information to put the results obtained in sections 5.3 and 5.4 together in this section. First, let us summarizing those results as follows: In supervised learning, when either the desired signal is the same as the input signal, or the desired signal is independent of the input signal, supervised learning degenerates to unsupervised learning. Then, let us study the results from the mutual information viewpoint. The mutual information between two random experiments $\alpha(p_1, p_2, \ldots, p_m)$ for event A and $\beta(q_1, q_2, \ldots, q_n)$ for event B is defined by [Jumarie 1990]

$$I(\beta|\alpha) = E\left(\log\frac{p(A_iB_j)}{p(A_i)p(B_j)}\right) \tag{173}$$

It is well known that $I(\beta|\alpha)$ reaches its maximum when the two events are the same and $I(\beta|\alpha)$ reaches its minimum zero when the two events are independent [Jumarie 1990], which are exactly the conditions for the supervised learning degenerating to unsupervised learning obtained in sections 5.3 and 5.4. Therefore, we can present a unified learning paradigm which includes both supervised and unsupervised learning. The system is shown in Figure 78. Notice that the system has the supervised structure since

supervised learning contains unsupervised as special cases. In Figure 78, two kind of external sources are defined: input source and desired source. If this two sources are the same or statistical independent of each other, the system runs in unsupervised mode, otherwise, it runs in supervised mode.

The major result of this chapter, therefore, can be stated as: In supervised learning, if the mutual information between the desired and input signals reaches the extremes, supervised learning degenerates to unsupervised learning.

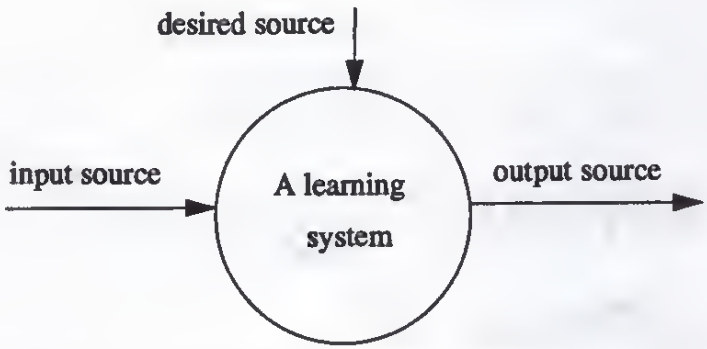


Figure 78. An unifying perspective of learning system

5.7 Discussions

We revealed a relationship between supervised MSE learning and unsupervised learning. We showed that supervised MSE learning defaults to unsupervised learning when the mutual information between the desired and input signals reaches it maximum or minimum. Based on this, we conclude that supervised structure can be viewed as a generalized structure for learning system. This observation is totally different from the conventional definition of unsupervised learning, and may be useful to design new hybrid learning system.

The results presented in this chapter can be viewed as a further step of Chapter 2. We also should stress that the temporal PCA network proposed in Chapter 3 can be included

in the unified perspective without any special assumption.

This chapter also showed that the role the input and desired signals played in a supervised learning is different, i.e., the input related correlation term is a anti-Hebbian learning and the desired related correlation term is a Hebbian learning.

A learning system, as pointed out in Chapter 1, is able to obtain information from the environment through all its external inputs. Traditionally, one thought that the larger the number of external signals, the more information can be provided by the environment. Based on the analysis given in this chapter, it is obvious that the amount of information the environment can provide does not depend necessarily on the number of information sources, but is also decided by the mutual information between external signals. However, we still don't know how to specify the network topology to extract optimally information from the environment (both input and desired signals). Likewise, we still don't know how to design the desired signal in order that a learning can get the most information from the environment. This is a very interesting problem because some preliminary experimental results indicated that different desired signals for a classification problem gave different results [Principe and Zahalka, 1993]. We will propose these problems for the future research in Chapter 6.

CHAPTER 6

CONCLUSIONS AND FUTURE RESEARCH

6.1 A Recapitulation of the Research

The primary goal of this research was to develop a unified perspective for adaptive learning systems. To achieve this goal, we analyzed various paradigms such as Hebbian learning, PCA learning, competitive learning, backpropagation algorithms for multilayer perceptrons, real time recurrent learning, higher order learning, and demonstrated the common factors in these algorithms. We found that correlation learning, which was thought to be very important in unsupervised learning, also plays a critical role in supervised learning. We pointed out in Chapter 2 that, supervised learning with the mean square error criterion as a cost function and gradient information as an optimization approach, the adaptation rules of the weights are a linear combination of several correlation operations. Chapter 2 also presents the reason why the linear operation is valid for gradient descent learning, since the dual network in a supervised system is a linear network with time-varying coefficients.

In order to make up for the lack of temporal extension to unsupervised learning, which is a necessary component of our unified perspective for learning systems, we switched to the study of temporal unsupervised systems. The temporal Principal Components Analysis (PCA) network is presented in Chapter 3. First, we proposed a temporal network for temporal PCA by including a short term memory mechanism in the topology. This approach is extensively used in supervised learning. Then, we adopted the eigenfunc-

tion decomposition approach to process random variables and random sequences to analyze the structure we proposed. We showed that the temporal PCA network trained with Hebbian rule provides not only the optimal orthogonal decomposition of the input signal, but also time and frequency information for stationary signals. In this chapter, we also pointed out the relationship between the temporal PCA network and eigenfilters in a filter bank arrangement. Examples of using temporal networks were given in Chapter 3. One showed the time-frequency information extraction by the temporal network; the other employed the temporal PCA network for multi-resolution analysis; and another one applied the temporal PCA network as a component in the transform domain Least Mean Square (LMS) filter, which was shown to have better convergence rate, less computation complexity, and on-line processing capability. It is straightforward to see that the training algorithm of the temporal PCA network belongs to the correlation learning principle.

Chapter 4 is actually an integration of the theory developed in previous chapters. The correlation relationship, the temporal structure for unsupervised PCA learning, and the relation between supervised and unsupervised ideas were applied to a novel structure with a new learning algorithm. The proposed approach was used to compute the crosscorrelation between two signals. We showed that this structure is necessary in order to get complete information about the crosscorrelation matrix between two signals. In addition, we extend the structure to several more powerful structures by using the Gamma memory and the nonlinear units in the network. Then, the network was applied to a still unsolved problem---the blind source separation problem. Experimental results of the advantages of the novel structure over other methods were given.

Chapter 5 went back to the unified perspective of learning systems. A very important relationship between supervised and unsupervised learning was described in this chapter. We showed that supervised learning defaults to unsupervised learning when the mutual information between the desired signal and the input signal reaches its extremes (minimum or maximum). Under this framework, supervised learning and unsupervised learning

systems share the same learning algorithms and network structures. Therefore, it is a very useful perspective which did not exit in the conventional distinction between them.

The importance of this research can be summarized as follows:

(1) The unified perspective highlights the common factors that most learning systems share. Hence, it is useful for an in-depth understanding of the difference between learning systems.

(2) The unified perspective points out the role that each signal plays in a learning system, so that it serves as a basis to design novel learning algorithms.

(3) The unified perspective studies the supervised and unsupervised learning under the same structure, therefore, proposes a novel approach to integrate supervised and unsupervised learning into one network.

(4) The unified perspective provides some biological evidence for the gradient based learning, since unsupervised learning has been proved biological plausible and shares the same learning properties with supervised learning.

6.2 Future Research Directions

As far as the future work goes, several issues should be pursued for the deep analysis of the unified perspective developed in this thesis:

(1) The supervised version of the competitive learning provides a way to analyze mathematically the performance of the competitive learning, since similar idea was used in statistical filters such median, rank filter and microstatistical filters in signal processing.

(2) For the temporal PCA network, more mathematical analysis is needed for the case of nonstationary signals. Recent achievements in wavelet theory may help us to conduct the analysis for nonstationary signals, since unitary transformation which is very similar to the transformation used in temporal PCA has been studied in the representation of nonstationary signals. But the difficulty lies in the fixed nature of wavelet bases, while the bases in

temporal PCA are adapted on-line to track the change of second-order statistics of the signal. Hence, it is very important to discuss the relationship between the temporal PCA and wavelets, and benefits to both research areas will result.

(3) In the application of the crosscorrelation estimation procedure proposed in Chapter 4 to the blind sources separation, we saw from the learning curve that a smaller crosscorrelation coefficient does not give the better separation results. So, a better criterion is required which gives a consistent result between the learning curve and the experimental results.

(4) Blind sources separation, or more generally by blind deconvolution problem, is a very important topic for digital signal processing, communication, and biology. It can be used for speech enhancement [Wang et al, 1996], channel equalization, noise and echo cancellation, and biological signal decomposition and detection. Recently, lots of research are focusing on it. Since one of the inherent advantages of neural networks is the nonlinearity, and nonlinearity may play an important role in blind deconvolution [Haykin, 1996], the neural approach is very promising.

(5) Since the main goal of the unified perspective is to help us understand the mechanism of learning, how to utilize the unified perspective developed in this research is important. Although an example was given in the thesis, I believe more novel algorithms and applications should be pursued in the future.

(6) As mentioned in section 5.7, the problem of how to design a desired signal for the sake of getting the most information for a learning system is worth pursuing. We may revisit the work done by Plumbley and Linsker [Plumbley and Fallside, 1989; Linsker, 1989] and analyze how the networks transfer information from input to output. Based on our understanding, this problem may be viewed as a constrained optimization problem described as follows:

$$\begin{aligned} & maxI(w, \mathcal{E}) \\ & st: \ neural \ topology \end{aligned} \tag{174}$$

where I denotes the mutual information, ε represents a variable who contains the information from the environment. The constrain "neural topology" is set for different neural network. It is well known that topology is associated with information capacity [Baum and Haussler, 1989]. But this is not a easy problem by any means, and it may take a long time to formulate. In summary, the research presented in the thesis brings several very promising directions in both theoretical and practical aspects.

REFERENCES

Allen, J., and Rabiner, L., "A unified approach to short-time Fourier analysis and synthesis," Proc. of IEEE, Vol. 65. No. 11, pp. 256-234, 1977.

Baldi, P., "Gradient descent learning algorithm overview: a general dynamical systems perspective, IEEE Trans. Neural Networks, Vol. 6, No. 1, pp. 182-195, 1995.

Baldi, P., and Hornik, K., "Neural networks and principal component analysis: learning from examples without local minima," Neural Networks, Vol. 2, No. 1, pp. 53-58, 1989.

Bannour, S., and Azimi-Sadjadi, M., "Principal components extraction using recursive least square learning," IEEE Trans. on Neural Networks, Vol.6, No.2, pp. 457-469, 1995.

Barlow H. B., "Unsupervised learning," Neural Computation, Vol. 2, No. 1, pp. 295-311, 1989.

Baum, E., and Haussler, D., "What size net gives valid generalization," Neural Computation Vol. 1, No. 2, pp. 151-160, 1989.

Bell, A., and Sejnowski, T., "An information-maximization approach to blind separation and blind deconvolution," Proc. of Neural Information Processing Systems (NIPS) Vol. 7, Denver, pp. 1011-1020, Cambridge, MA, MIT Press, 1995.

Bellanger, M., Adaptive digital filters and signal analysis, M. Dekker, New York, 1987.

Beyerbach, D., and Nawab, H., "Principal components analysis of the short-time Fourier transform," Proc. of International Conference on Acoustics, Speech, and Signal Processing (ICASSP), Vol. 2, pp. 1725-1728, IEEE, Piscataway, NJ, 1991.

Bourlard, H., and Kamp, Y., "Auto-association by multilayer perceptrons and singular value decomposition," Biol. Cybern., Vol. 59, No. 3, pp. 291-299, 1988.

Brillinger, D., Time series: Data analysis and theory, Expanded edition, Holden-Day, Inc., San Francisco, CA, 1980.

Chappell, G., and Taylor, J. G. "Temporal Kohonen map," Neural Networks, Vol. 6

No. 3, pp 441-445, 1993.

Cohen, L., Time-frequency analysis, Prentice-Hall, Englewood Cliffs, NJ, 1995.

Comon, P., "Independent component analysis: A new concept?," Signal Processing, Vol. 36, No.3, pp. 287-314, 1994.

Darken, C., and Moody, J., "Towards faster stochastic gradient search," Proc. of Neural Information Processing Systems (NIPS) 4, pp. 1009-1016, San Mateo, CA: Morgan Kaufmann, 1992.

Daubechies, I., Ten lectures on wavelets, SIAM, Philadelphia, 1992.

Dautrich, B., and Rabiner, L., "On the effects of varying filter banks parameters on Isolated word recognition," IEEE Trans. on ASSP, Vol. 31, No. 4, pp. 878-886, 1983.

Davidson, G., and Falconer, D., "Reduced complexity echo cancellation using orthonormal functions," IEEE Trans. on Circuits and Systems, Vol.38, No.1, pp. 20-28, 1991.

Diamantaras, K. L., and Kung, S. Y., "Cross-correlation neural networks models," IEEE Trans. on Signal Processing, Vol. 42, No. 11, pp. 3218-3223, 1994.

Duda R., and Hart P., Pattern classification and scene analysis, Wiley, New York, 1973.

Elfadel, I. M., "Global dynamics in principal singular subspace networks," Proc. of International Conference on Acoustics, Speech, and Signal Processing (ICASSP), Vol. 5, pp. 3371-3374, IEEE, Piscataway, NJ, 1995.

Feller, W, An Introduction to probability theory and its applications, Vol. 1, 3rd ed., New York, Wiley, 1968.

Foldiak, P., "Adaptive network for optimal linear feature extraction," Int. Joint Conf. on Neural Networks, Vol. 1, pp.401-405, Washington, DC, 1989.

Fukunaga, K., Statistical pattern recognition, 2nd ed. Academic Press, San Diego, CA, 1990.

Gabor, D., "Theory of communication," J. EE, Vol. 93, No. 4, pp. 429-457, 1946.

Gerbrands, J., "On the relationships between SVD, KLT, and PCA," Pattern Recognition Vol.14, No.1, pp. 78-89, 1 981.

Giles, C. Lee, and Maxwell, T., "Learning, invariance, and generalization in higher-order neural networks," Applied Optics, Vol. 26, No. 23, pp. 435-443, 1987.

Giles, C., Lee, Horne, B.G., and Lin, T., "Learning and extraction finite state automata with second-order recurrent neural networks," Neural Computation, Vol. 5, No. 2, pp. 293-307, 1992.

Gish H., "A probabilistic of artificial neural networks," Proc. of International Conference on Acoustics, Speech, and Signal Processing (ICASSP), Vol. 3, pp. 2135-2138, IEEE, Piscataway, NJ, 1990.

Hampshire J., and Perlmutter, B., "Equivalence proofs for multilayer perceptron classifiers and Bayesian discriminant function," Proc. of the 1990 Connectionist Model Summer School, D. Touretzky, J. Elman, T. Seinowski, and G. Hinton Eds. Morgan Kaufmann, San Mateo, CA. 1990

Haykin, S., Adaptive filter theory, 2nd ed., Prentice-Hall, Englewood Cliffs, NJ, 1991.

Haykin, S, Neural Networks---A comprehensive foundation, Macmillan College Publishing Company, New York, 1994.

Haykin, S., Adaptive filter theory, 3nd ed. Prentice-Hall, Englewood Cliffs, NJ, 1996.

Hebb, D. O., The organization of behavior: A neuropsychological theory, Wiley, New York, 1949.

Hecht-Nielsen, R., "Counterpropagation networks," Applied Optics, Vol. 26, No. 11, pp. 4979-4984, 1987.

Hecht-Nielsen, R., "Applications of counterpropagation networks," Neural Networks Vol.1, No. 1, pp. 131-139, 1988.

Hertz J., Krogh A., and Palmer R. G., "Introduction to the theory of neural computation," Addison-Wesley, Reading, MA, 1991.

Hinton G. E., "Connectionist learning procedure," In machine learning: paradigms and methods, J. of Artificial Intelligence, Vol.13, No. 2, pp. 287-302, 1989.

Jumarie G., Relative information, Springer-Verlag, Berlin, Heidelberg, New York, 1990.

Jutten, K., and Herault, J., "Blind separation of sources, Part I: An adaptive algorithm based on neuromimetic architecture," Signal processing, Vol. 24, No. 1, pp. 1-10, 1991.

Kapur, J., and Kesavan, H., Entropy optimization principles with applications, Aca-

demic Press, Boston, 1992.

Karhunen, J., and J. Joutsensalo, "Representation and separation of signals using nonlinear PCA type learning," Neural Networks, Vol. 7, No. 1, pp. 113-127, 1994.

Karhunen, J., and J. Joutsensalo, "Generalizations of principal components analysis, optimization problems, and neural networks," Neural Networks, Vol. 8, No. 4, pp. 549-562, 1995.

Kohonen, T., "Self-organizing formation of topologically correct feature maps," Biological Cybernetics, Vol. 43, No. 1, pp. 59-69, 1982.

Kohonen, T., Self-organization and associative memory, Springer-Verlag, Berlin, 1989.

Kohonen, T., "The self-organizing map," Proc. of IEEE Vol. 78, No. 9, pp. 1464-1480, 1990.

Kosko, B., "Unsupervised learning in noise," IEEE Trans. Neural Networks, Vol. 1, No. 1, pp. 278-286, 1990.

Kullback S., Information theory and statistics. John Wiley & Sons, Inc. New York, 1954.

Kung, S. Y., and Diamantaras, K. I., "A neural network learning algorithm for adaptive principal component extraction(APEX)," Proc. of International Conference on Acoustics, Speech, and Signal Processing (ICASSP), Vol. 2, pp. 861-864, Albuquerque, NM, 1990.

Lee, Y., Giles, C. Lee, and Maxwell, T., "Machine learning using a higher-order correlation network," Physica 22D, Vol. 87, No.3, pp. 1011-1019, 1986.

Linsker, R., "From basic network principles to neural architecture,"(series), Proc. Nat'l Academic of Sciences USA, Vol. 83, No. 12, pp. 7899-7908, 1986.

Linsker, R., "Self-organization in a perceptual network," IEEE Computer, Vol. 56, No. 9, pp. 1324-1331, 1988.

Liu, S. C., "Time-varying spectra and linear transformation," The Bell System Technical Journal, Vol. 50, No. 7, pp. 1052-1067, 1971.

Makhoul, J., "On the eigenvectors of symmetric Toeplitz matrices," IEEE Trans. On ASSP, Vol. 29, No. 4, pp. 767-772, 1981.

Makhoul, J., "Pattern recognition properties of neural networks," IEEE Neural Networks Workshop on Signal Processing, Vol.1, pp. 173-187, 1991.

Matsuoka, K., and Kawamoto, M., "A neural network that self-organizes to perform three operations related to principal component analysis," Neural Networks, Vol. 7, No. 5, pp. 753-765, 1994.

Matsuoka, K., Ohya, M., and Kawamoto, M., "A neural net for blind separation of nonstationary signals," Neural Networks, Vol. 8, No.3, pp. 411-419, 1995.

Mendel, J., Adaptive, learning, and pattern recognition systems: Theory and applications, Academic Press, New York, 1970.

Mendel, J., "Tutorial on higher-order statistics (spectra) in signal processing and system theory: Theoretical results and some applications," IEEE Proc., Vol. 79, No. 3, pp. 278-305, 1991.

Nadal J.-P., and Parga N., "Duality between learning machines: a bridge between supervised and unsupervised learning," Neural Computation, Vol. 6, No. 3, pp. 491-508, 1994.

Narayan, S., Peterson, A., and Narasimha, M., "Transform domain LMS algorithm," IEEE Trans. on ASSP, Vol.31, No. 3, pp. 1024-1031, 1983.

Nawab, H., and Beyerbach, D., "Principal decomposition of time-frequency distribution," IEEE Trans. SP, Vol. 41, No. 11, pp. 3182-3186, 1993.

Nikias, C. L., and Mendel, J. M., "Signal processing with higher-order spectra," IEEE Signal Processing Magazine, Vol. 12, No. 3, pp. 14-32, 1993.

Oja, E., "A simplified neuron model as a principal component analyzer," Journal of Mathematical Biology Vol. 15, No. 3, pp. 167-173, 1982.

Oja, E., "Principal components, minor components, and linear neural networks," Neural Networks, Vol. 5, No. 3, pp. 927-936, 1992.

Oja, E., and Karhunen, J., "Nonlinear PCA: algorithms and applications," TR-A18, Helsinki Univ. of Tech., Lab. of Computer and Information Science, 1993.

Oppenheim, A., and Schafer, R., Discrete-time signal processing, Prentice-Hall, Englewood Cliffs, NJ, 1989.

Palmieri, F., Zhu J., and Chang C., "Anti-hebbian learning in topologically constrained linear networks: A tutorial," IEEE Trans. on Neural Networks, Vol. 4, No. 5, pp. 748-761, 1993.

Papoulis, A. Probability, random variables, and stochastic processes, McGraw-Hill, New York, 1991.

Picinbono, B., Random signals and systems, Prentice-Hall, Englewood Cliffs, NJ, 1993.

Pineda, F., "Recurrent backpropagation and the dynamical approach to adaptive neural computation," Neural Computation, Vol. 1. No. 4, pp. 280-290, 1989.

Plumbley, M. D., and Fallside, F., "An information-theoretic approach to unsupervised connectionist models," In D. Touretzky et al. Eds., Proc. of the 1988 Connectionist Models Summer School, San Mateo. CA:Morgan-Kaufmann, 1988.

Plumbley, M. D., "Efficient information transfer and anti-Hebbian neural networks," Neural Networks, Vol. 6, No. 6, pp. 823-833, 1993.

Plumbley, M. D., "Lyapunov functions for convergence of the principal components algorithms," Neural Networks, Vol. 8, No. 1, pp. 11-23, 1995.

Principe, J. C., de Vries, B., and de Oliveira, P., "The gamma filter - a new class of adaptive IIR filters with restricted feedback," IEEE Transactions on Signal Processing, vol. 41, no. 2, pp. 649-656, 1993

Principe, J. C., and Zahalka, A., "Transient Detection Using Neural Networks: The search for the Desired Signal," Proc. of Neural Information processing Systems (NIPs), Vol. 5, pp. 688-695, 1993.

Principe, J. C., Fundamentals of neural networks, Prentice-Hall, Englewood Cliffs, NJ, to be published in 1996.

Rabiner, L., and Gold, B., Theory and application of digital signal processing, Prentice-Hall, Englewood Cliffs, NJ, 1975.

Rabiner, L., and Schafer, R. W., Digital processing of speech signal, Prentice-Hall, Englewood Cliffs, 1978.

Richard M., and Lippmann R. P., "Neural network classifiers estimate Bayesian a posteriori probability," Neural Computation, Vol. 3, No. 3, pp. 461-483, 1991.

Ritter, H., Martinetz, T., and Schulten, K., Neural computation and self-organizing maps: An introduction, Addison-Wesley, Reading, MA, 1992.

Ritter, H., and Schulten, K., "Convergence properties of Kohonen's topology conserving maps: Fluctuations, stability, and dimension selection, Biological Cybernetics, Vol. 60, No. 1, pp. 59-71, 1986.

Rubner, J., and Tavan, P., "A Self-organizing network for principal component analysis," Europhysics Letters Vol.10, No. 22, pp. 693-698, 1989.

Rumelhart, D. E., McClelland, J., and the PDP Research Group, Parallel distributed processing: Explorations in the microstructure of cognition, Vol. 1, MIT Press, Cambridge, 1986

Rumehalt, D., and Zipser, D., "Feature Discovery by Competitive Learning," Cognitive Science Vol. 9, pp. 75-112, 1985.

Sanger, T. D., "Optimal unsupervised learning in a single-layer linear feedforward neural network," Neural Networks, Vol. 2, No.3, pp. 459-473, 1989.

Schafer, R. W., Rabiner, L., and Herrmann, O., "FIR digital filter banks for speech analysis," The Bell System Technical Journal, Vol. 54, No. 3, pp. 1055-1079, 1975.

Scharf, L., Statistical signal processing, Addison-Wesley, Reading, MA, 1991.

Shannon, C., and Weaver, W., The mathematical theory of communication, The University of Illinois Press, Urbana, IL, 1949.

Shynk, J., "Frequency-domain and multirate adaptive filtering," IEEE SP Magazine, Vol. 11, pp. 22-34, 1992.

Solla, S. A., Levin, E., and Fisher, M., "Accelerated learning in layered neural networks," Complex Systems, Vol. 2, No. 5, pp. 625-640, 1989.

Strum, R. D., and Kirk, D. E., First principles of discrete systems and digital signal processing, Addison-Wesley, Reading, MA, 1988.

Swami, A., and Mendel, J., "Time and lag recursive computation of cumulants from state-space model," IEEE Trans. AC. Vol. 35, No. 1, pp. 67-77, 1990.

Takens, F., Lecture notes in mathematics, Vol. 898, Springer, Berlin, 1981.

Therrien, C., Discrete random signal and statistical signal processing, Prentice-Hall, Englewood Cliffs, NJ 1992.

Tracey, J., and Principe, J. C., "Isolated speech recognition with focused gamma networks," Proc. World Congress on Neural Networks, Vol. 3, pp. 87-90, 1993.

Tsypkin, I., Adaptation and learning in automatic systems, Academic Press, New York, 1971.

Vaidyanathan, P. P., Multirate systems and filter banks, Prentice-Hall, Englewood Cliffs, NJ, 1993.

Van Grevan, S., and Van Compernolle, D., "Signal separation by symmetric adaptive

decorrelation: stability, convergence, and uniqueness," IEEE Trans. SP. Vol. 43, No. 7, pp. 2891-2899, 1995.

Van Trees, H. L., Detection, estimation, and modulation theory, Part I, Wiley, New York, 1968.

Vaz, F., and Principe, J. C., "Spike detection using neural networks," Internal report, CNEL Lab, University of Florida, 1994.

von Malsburg, C., "The correlation theory of the brain function," Physics of neural networks: Models of neural networks II, Eds. E. Domany, J. L. Hemmen, and K. Schulten, Springer-Verlag, 1994.

Wan, E., "Temporal backpropagation for FIR neural networks," International Joint Conf. on Neural Networks, San Diego, Vol. 1, pp. 575-580, 1990.

Wang, C., Kuo, J-M., and Principe, J. C., "Using anti-Hebbian training to find orthogonal signals," Proc. of International Symposium on Artificial Neural Networks (ISANN), Taiwan, pp. 608-613, 1994.

Wang, C., Kuo, Jyh-Ming, and Principe, Jose. C., "A relation between Hebbian and MSE learning," Proc. of International Conference on Acoustics, Speech, and Signal Processing (ICASSP), Vol. 5, pp. 3363-3366, 1995.

Wang, C., and Principe, J. C., "A relation between Hebbian and MSE learning in nonlinear networks," World Congress on Neural Networks, Washington D.C., pp. 540-543, 1995.

Wang, C., and Principe, J. C., "Optimal energy time-frequency analysis using PCA learning in time," Internal Report, CNEL Lab, University of Florida, 1996.

Wang, C., and Principe, J. C., "Learning dynamics using static network with correlation rule," Internal Report, CNEL Lab, University of Florida, 1996.

Wang, C., Wu, H-C, Xu, D., and Principe, J. C., "Simultaneous blind separation and noise cancellation for telephone-line conversation," Submitted to IASTED Intel conf. on Signal and Image Processing, April 1996.

Weinstein, E., Feder, M., and Oppenheim, A., "Multi-channel signal separation by deconvolution," IEEE Trans. Speech, Audio Processing, Vol. 1, No.4, pp. 405-413, 1993.

Werbos, P., "Backpropagation through time: What it does and how to do it," Proc. of IEEE, Vol. 78, No. 11, pp. 1550-1560, 1990.

White H., "Learning in artificial neural networks: A statistical perspective," Neural Computation, Vol. 1, No.2, pp. 425-464, 1989.

Widrow, B., Hoff, M., "Adaptive switching circuits," IRE WESCON Convention Record, pp. 96-104. 1960.

Widrow, B., and Walach, E., "On the efficiency of the LMS algorithm with nonstationary inputs," IEEE Trans. on Information Theory, Vol.30, No. 2, pp. 353-364, 1984.

Widrow, B., and Stearns, S. D., Adaptive signal processing, Prentice-Hall, Englewood Cliffs, NJ, 1985.

Williams, R. J., and Zipser, D., "A learning algorithm for continually running fully recurrent neural networks, "Neural Computation, Vol. 2, No. 2, pp. 490-501, 1990.

BIOGRAPHICAL SKETCH

Chuan Wang was born in Oct. 9, 1963, in Chongqing China. He received his Bachelor of Science degree in electrical engineering in 1985, and Master of Science in systems engineering in 1991, both from Jiaotong University, China. He also spent more than one year at the University of Texas at Dallas to study mathematics and statistics. He entered the graduate program in electrical engineering at the University of Florida at August 1992. Since then, he has worked with Dr. Jose Principe on various topics in signal processing. His main research interests are adaptive systems, speech enhancement and recognition, digital communication, and estimation theory.

I certify that I have read this study and that in my opinion it conforms to acceptable standards of scholarly presentation and is fully adequate, in score and quality, as a dissertation for the degree of Doctor of Philosophy.

Jose C. Principe, Chairman
Professor of Electrical and Computer Engineering

I certify that I have read this study and that in my opinion it conforms to acceptable standards of scholarly presentation and is fully adequate, in score and quality, as a dissertation for the degree of Doctor of Philosophy.

William W. Edmonson
Assistant Professor of Electrical and Computer Engineering

I certify that I have read this study and that in my opinion it conforms to acceptable standards of scholarly presentation and is fully adequate, in score and quality, as a dissertation for the degree of Doctor of Philosophy.

John G. Harris
Assistant Professor of Electrical and Computer Engineering

I certify that I have read this study and that in my opinion it conforms to acceptable standards of scholarly presentation and is fully adequate, in score and quality, as a dissertation for the degree of Doctor of Philosophy.

Jian Li
Assistant Professor of Electrical and Computer Engineering

I certify that I have read this study and that in my opinion it conforms to acceptable standards of scholarly presentation and is fully adequate, in score and quality, as a dissertation for the degree of Doctor of Philosophy.

Mark Yang
Professor of Statistics

This dissertation was submitted to the Graduate Faculty of the College of Engineering and to the Graduate School and was accepted as partial fulfillment of the requirements for the degree of Doctor of Philosophy.

August 1996

for Winfred M. Phillips
Dean, College of Engineering

Karen A. Holbrook
Dean, Graduate School